Real-Time Detection of Pedestrian Traffic Lights for Visually-Impaired People

Marcelo C. Ghilardi, Gabriel Simões, Jônatas Wehrmann, Isabel H. Manssour, and Rodrigo C. Barros
marcelo.ghilardi@acad.pucrs.br, gabriel.simoes.001@acad.pucrs.br, jonatas.wehrmann@acad.pucrs.br
isabel.manssour@pucrs.br, rodrigo.barros@pucrs.br
Escola Politécnica, Pontifícia Universidade Católica do Rio Grande do Sul

Abstract—**Nowadays there are more than 250 million visually-impaired people worldwide and mobility autonomy in outdoor environments is perhaps the greatest challenge they have to face. More specifically, crossing the street with no human aid is an open problem, since the majority of the pedestrian traffic lights in underdeveloped countries do not provide sound aids. There are very few studies addressing the detection of pedestrian traffic lights based on images acquired by mobile devices, and to the best of our knowledge there is a clear gap in the literature regarding the use of recent state-of-the-art computer vision approaches such as deep neural networks for addressing such a problem. In this paper, we investigate the current state-of-the-art in localization/detection and classification based on deep neural networks, and we present a solution for the detection of pedestrian traffic lights together with their current state for helping visually-impaired people to cross the streets with the aid of their mobile devices. For such, we provide a novel public dataset with 4,399 labeled images of pedestrian traffic lights and we present a detailed comparison among the state-of-the-art methods for classification and localization/detection. We show empirical evidence regarding the feasibility of embedding our approach in mobile devices so it can be used by people with visual impairment.**

I. Introduction

According to 2017 data from the World Health Organization[1], there are approximately 253 million visually-impaired people worldwide. Among them, 36 million are blind and 217 million have moderate to severe vision impairment. Blind tracks, white canes, and guide dogs are typically used to help visually-impaired people to walk outdoors [1]. Despite of those aids, mobility autonomy in outdoor environments still poses a big problem for visually-impaired people.

In this context, one of the major challenges daily faced by the visually impaired is to cross the street independently. A common solution for this problem is to associate a sound signal with the pedestrian traffic lights, though in several countries (e.g. third-world countries such as Brazil) the vast majority of the pedestrian traffic lights do not provide an associated aiding sound signal. Moreover, the American Council of the Blind[2] states that the so-called "cuckoo-chirp type signals" can lead to incorrect decisions due to the difficulty of identifying, in corners, which street is allowed for crossing. When multiple tones are used, there is the difficulty in remembering which tone is used for each direction.

[1]http://www.who.int/mediacentre/factsheets/fs282/en
[2]http://acb.org/

Several solutions for the detection of traffic lights for autonomous vehicles can be found in the literature [2]–[7], though very few of them were developed for the detection of pedestrian traffic lights based on images acquired by mobile devices [8]–[11]. In addition, to the best of our knowledge, there is a gap in the pedestrian traffic lights literature regarding more recent state-of-the-art computer vision techniques such as deep neural networks.

Hence, our main goal in this work is to present a solution for the detection of pedestrian traffic lights and their current state for helping visually-impaired people to cross the streets with the aid of their mobile devices. For such, we carefully investigate the current state-of-the-art in localization/detection and classification based on deep neural networks, and the feasibility of embedding those models in a mobile device.

Our main contributions include: (1) the availability of a novel public dataset with 4,399 labeled images of pedestrians traffic lights; (2) a comparison among the deep learning state-of-the-art for classification and detection, namely YOLO [12], Faster R-CNN [13], and SSD [14]; (3) a thorough empirical analysis that shows the feasibility of our approach to be embedded in a mobile device and thus be used by people with any kind of visual impairment.

The remainder of this paper is organized as follows. We discuss related work on traffic lights and visually-impaired aid approaches in Section II. We describe our novel dataset and the object detection methods we make use in Sections III and IV, respectively. In Section V we discuss our results and main findings regarding the suitability of employing state-of-the-art deep neural approaches for pedestrian traffic light detection and classification. Finally, we end this paper with our conclusions and future work directions in Section VI.

II. Related Work

In the past couple of years, mostly due to the fast advances in autonomous vehicles research, several studies have focused in developing novel methods for detection of traffic lights specially for cars [2]–[7], [10], [15]. We can broadly classify the related work into the following categories: i) handcrafted-features based techniques; and ii) representation-learning based techniques, a.k.a. deep learning approaches. Whereas most of those papers rely on detecting traffic lights for autonomous navigation systems, only a small portion of them aim to help visually-impaired people by detecting pedestrian traffic lights [8], [11], [16], [17].

A. *Traffic Lights Detection for Vehicles*

Li et al. [2] developed a system for traffic light detection for cars that makes use of visual information from an on-vehicle camera. They collect and leverage prior information such as aspect ratio, area, location, and context of traffic lights, along with some methods to improve accuracy – e.g., using an inter-frame analysis with feature learning algorithms. Authors claim competitive results on the VIVA dataset [18]. Liu et al. [3] proposed a method that makes use of Extreme Learning Machines [19] trained over HOG (Histogram of Oriented Gradients) [20] and LBP (Local Binary Pattern) [21] features. The authors were capable of running that approach in real-time in smartphones by using a CPU-GPU fusion approach. They report fast running-time while keeping competitive predictive results. A deep learning approach proposed in [5] uses stereo video sequences and vehicle odometry to detect traffic lights in real-time. The authors introduced a labeled dataset with 5,000 images of traffic lights that present controlled variation of lighting and weather (i.e., sunny to light rain). Probably the major drawback of their approach is the very use of stereo images and odometry, which makes it quite difficult to employ their method in regular smartphones for detecting pedestrian traffic lights. Jensen et al. [6] proposed the use of the well-known YOLO (You Only Look Once) [12] approach for real-time detection of traffic lights. They performed experiments on the LISA [22] dataset. In addition, Lee and Park [7] also employed that dataset along with deep neural networks. Nevertheless, they focus on a pre-processing phase for color segmentation and further application of SVMs to reduce false regions from the segmentation step.

We have found two papers [10], [15] that explicitly aim to help visually-impaired people. Even though [10] focus in running on mobile devices, both of them are designed to detect traffic lights for vehicles using traditional handcrafted-features based techniques to extract shape and color information. In addition, the work in [9] proposes the use of traffic light detectors to improve the accessibility of color-blind people.

B. *Pedestrian Traffic Lights Detection*

Roters et al. [8] proposed a system for mobile devices based on handcrafted-features based approaches for pedestrian traffic lights detection. They have introduced a novel dataset with images of pedestrian traffic lights from Germany. Cheng et al. [11] improved that dataset, and also presented an approach that employs SVMs trained with HOG features. Overall, they demonstrated fair precision-recall values and proper running-time performance when using a portable computer with an RGB camera. The work in [16] presents a vision system for traffic information detection through a smartphone camera to help the visually impaired. One of its features is the recognition of traffic lights through a hybrid adaptive boosting (Adaboost) [23] and template matching algorithm. However, the authors do not make their dataset publicly-available for evaluation. A different approach based on integration of multiple services to build a model was presented by Alghamdi et al. [17]. One of those services is the pedestrian traffic light detection that was developed based on SURF (Speeded-Up Robust Features) [24]. They present a brief evaluation but also do not make their dataset available for other researchers. Mascetti et al. [25], [26] were concerned with a different approach: the development of a robust method for image acquisition. They used traditional handcrafted-features algorithms focusing on solving the problem of underexposed traffic lights.

As described herein, there are few studies for traffic light detection for cars that make use of deep learning [5]–[7]. Nevertheless, despite the existence of papers on real-time pedestrian traffic light detection [8], [11], [16], [17], none of them employ state-of-the-art approaches such as deep neural networks and analyze the feasibility of running the models within a smartphone. While there are public datasets for car traffic lights [18], [22] that are widely used, there are very few datasets [8], [11] with pedestrian traffic lights images. Moreover, the amount of images available for training in those datasets are probably insufficient for deep neural networks to properly learn and generalize . For instance, the dataset in [8] comprises only 300 images from the German standard for training. The second dataset [11] includes only 13 video-sequences from Italian and Chinese pedestrian traffic lights.

III. A Novel Pedestrian Traffic Light Dataset (PTLD)

Our research on traffic light detection led us to papers that mostly deal with traffic lights for cars, and hence provide datasets that contain such images. We found only two public datasets ([8], [11]) that contain pedestrian traffic lights, both with insufficient amount of images for deep neural networks to learn and properly generalize.

Hence, we decided to develop a novel pedestrian traffic light dataset and make it publicly available. We call it the Pedestrian Traffic Light Dataset (PTLD), and it can be downloaded at https://tinyurl.com/y9z7xodw.

PTLD was created following a rigorous protocol. First, with the collaboration of several people and using multiple smartphone models, we collected a variety of videos and images with and without pedestrian traffic lights from different Brazilian cities, with lights indicating allowance or prohibition to cross the street. In Brazil, there is a variety of shapes that pedestrian traffic lights can assume, and sometimes in the very same city the shapes may vary. Therefore, as shown in Figure 1, we were capable of collecting images for PTLD that have distinct image quality, lighting conditions, size, angles, and shapes.

After the video collection period, we extracted two images per second from each collected video, and added them to PTLD. We kept the original size of all images and we did not apply any kind of filters for image enhancement. We did use a Haar Cascade-based algorithm [27] to blur the faces of people that eventually appeared in the images in order to keep them in anonymity.

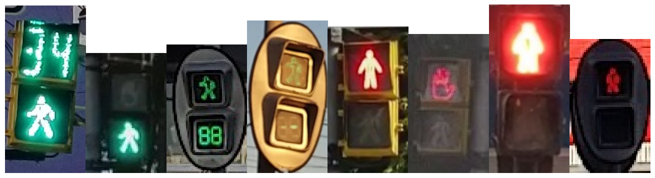


Fig. 1. Examples of images in the PTLD dataset with different resolutions, lighting conditions, sizes, angles, and shapes.

Finally, we manually annotate all images using a tool developed by ourselves, which is also publicly-available[3], allowing the identification and classification of pedestrian traffic lights. Our tool generates a textual annotation file for each image and also offers the option to export to many well-known annotations formats such as the XML Pascal-VOC format that is employed by one of the deep learning methods we evaluated in this work.

PTLD comprises a total of 4,399 images, 4,286 of them annotated with the current state of 4,645 traffic lights (allowance or prohibition to cross the street), and 128 images with no traffic light whatsoever. We divide all images into three sets: training, validation, and testing. The training set contains 3,443 images, whereas the validation and test sets contain 478 images each, as presented in Table I.

TABLE I
IMAGES DISTRIBUTION IN THE PTLD.

Class	**Training**	**Validation**	**Testing**
Go	913	201	240
Stop	2,219	236	249
Off	501	51	35
No traffic lights	113	12	3

PTLD comprises three actual classes: go, stop and off. The *Go* class is when the traffic light is open for the pedestrian to cross the street. It comprises images with white or green lights in varied shapes. The *Stop* class is when the traffic light prohibits the pedestrian to cross the street, usually signaled by a red light in diverse shapes. The *Off* class is when the lights are out, and it usually occurs intermittently between the *Go* and *Stop* classes.

IV. Object Detection Methods

Convolutional Neural Networks (ConvNets) [28] have been used for image classification and object recognition tasks for achieving state-of-the-art results [29], [30]. A ConvNet is a *deep learning* [31] approach that is capable of learning representations from raw data, and it has been applied with success to tasks such as image classification [32]–[35], object detection [13], [36], video understanding [37], [38], content-based retrieval [39], and text classification [40], [41].

Since ConvNets learn data-driven models, their performance largely depends on available labeled datasets. For achieving human-level performance, these datasets should ideally contain thousands or millions of instances. For instance, Imagenet [42] is the standard dataset for building image classification models due the amount of classes and instances available. For the task of object detection, both PASCAL VOC [43] and MS COCO [44] are often used. They comprise annotated objects that belong from 20 to 80 different classes.

In this work, we evaluate state-of-the-art ConvNet-based methods for object detection and classification in our Pedestrian Traffic Light Dataset (PTLD). The following approaches were used in our experiments: (i) Faster R-CNN; (ii) YOLO Full; (iii) YOLO Tiny; and (iv) SSD. We describe them next.

A. *Faster R-CNN*

Faster R-CNN [13] comprises a Region Proposal Network (RPN), which is a fully-convolutional network that predicts both the bounding box position and the scores at each position. RPN is trained in an end-to-end fashion to generate high-quality region proposals that are used by Fast R-CNN [45] for detection. RPN and Fast R-CNN are merged into a single network sharing some convolutional layers, resulting in Faster R-CNN, which is more efficient and effective than its precursor.

Figure 2 illustrates the Faster R-CNN overall architecture. It is comprised of two modules. The first module is the ConvNet that proposes regions (RPN), and the second is the Fast R-CNN detector, which learns from the proposed regions. Given an input image, it outputs a set of region proposals along with a score. Such a score indicates whether the given region proposal belongs to a given class or to background.

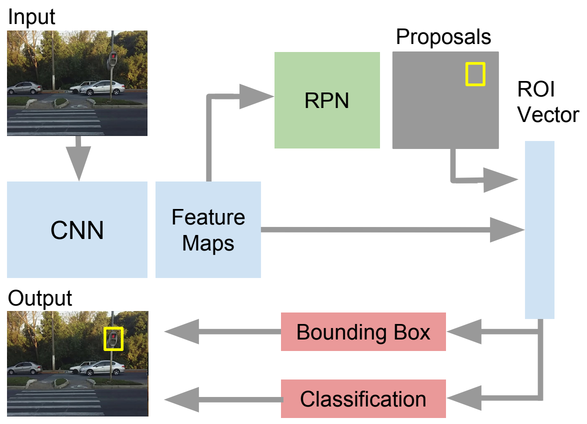


Fig. 2. Faster R-CNN pipeline.

B. *YOLO*

You Only Look Once (YOLO) [36] is another ConvNet-based object detection approach that transforms object detection into a regression problem. The YOLO pipeline is straightforward from the input image to bounding box coordinates and class probabilities via a ConvNet. Such a pipeline can detect objects with similar speed of a classification pipeline. Figure 3 depicts the YOLO pipeline, where convolutional

[3]https://github.com/kabrau/PyImageRoi

output features are used to predict both a set of bounding boxes and class probabilities distributed within a grid. Each grid cell predicts a set of bounding boxes and confidence scores. These scores reflect the model's confidence that such a cell contains an object. Also, those confidence scores threshold the bounding box on whether they are providing a valid detection. The simple and straightforward pipeline also contributes to an improved time performance, making the system capable of detecting objects in 45 FPS when running over an NVIDIA® Titan X GPU.

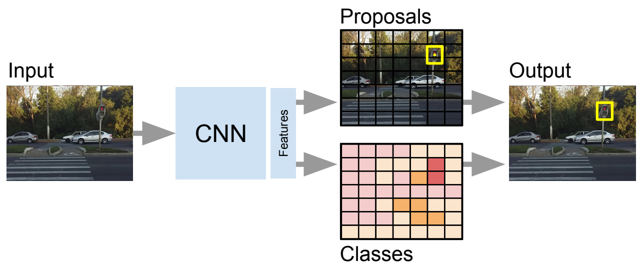


Fig. 3. YOLO pipeline.

An improvement of YOLO, namely YOLO V2 [12], excludes the fully-connected output layer of the original ConvNet architecture, making YOLO V2 flexible to variable-size image inputs. It also includes batch normalization to help regularizing the model, and residual connections in order to better locate small objects (the so-called "fine-grained features").

C. SSD

Single Shot MultiBox Detector (SSD) [14] is an object detection method that shares similarities with YOLO. For instance, SSD also detects objects with a single ConvNet forward pass per image. It detaches the resulting bounding boxes into a set of default boxes with different scales and ratios. The prediction output generates scores for the presence of each object class in each default box and also adjustments to make the box better fit the object. However, differently from the original YOLO architecture, SSD combines predictions from multiple feature maps with different resolutions, allegedly better handling objects of various sizes. SSD's single pass pipeline provides impressive prediction time that can reach 59 FPS with a GPU like NVIDIA® Titan X. Figure 4 illustrates the SSD pipeline and the resulting composition of different-scale feature maps predictions.

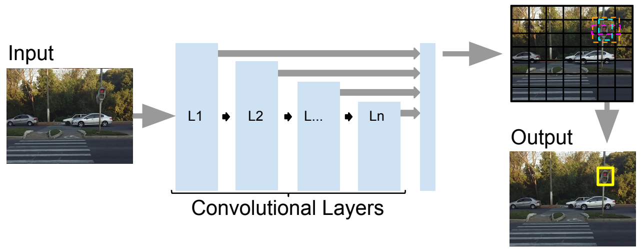


Fig. 4. SSD pipeline.

V. Experimental Analysis

To validate the use of PTLD to train accurate models for pedestrian traffic lights detection, we execute experiments with the state-of-the-art deep learning based approaches: YOLO [36], SSD [14], and Faster R-CNN [13]. Our experimental environment was provided by a Intel® Xeon® E5-2603 and an NVIDIA® GTX 1080TI GPU with 11GB of memory. We trained each method for 4 days. All networks were pre-trained on Imagenet [42], which helps for a better weights' initialization and improves learning with the transference of patterns present in a large classification dataset. In addition to the pre-training, we apply the same data augmentation strategy for all models. Our data augmentation comprises random crops of 90% of the image area and horizontal flips. As described in the PASCAL-VOC Challenge [43], our evaluation criteria consider correct predictions those with intersection over union (IOU) regarding the predicted box and the ground-truth above 0.5. A summary of the experiments performed over PTLD can be seen in Table II.

TABLE II
Summary of PTLD experiments.

Method	mAP Val.	mAP Test	Batch	Total iter.	Best iter.
Faster R-CNN	74.0	77.8	1	200,000	200,000
SSD	73.6	**87.9**	24	200,000	72,325
YOLO Full	**74.5**	76.2	64	100,000	30,000
YOLO Tiny	44.7	35.8	64	100,000	40,000

A. Faster R-CNN

For training a Faster R-CNN model, we adopt an *Inception Resnet-v2* [46] architecture, which comprises 59,419,965 parameters and 475.1MB of storage space for each model. The model was fine-tuned from an Imagenet [42] pre-trained model. We use the validation set to observe the training evolution. Due to restrictions of the method, the batch size is set to 1 image. Momentum is set to 0.9, and the learning rate is kept fixed in 4×10^{-4}.

According to the validation data, the best Faster R-CNN model was created at iteration 200,000, which provides a mAP=74.0. After evaluating such a model in the test set, the resulting mAP=77.8, which is the second best mAP of all experiments. Even though it provides the second best mAP, Faster R-CNN is the slowest method to train. It reached 200,000 iterations in the 4 days of training time.

B. SSD

We also train an SSD [14] network using PTLD. We adopt a *Resnet-v2* architecture, which was previously trained on Imagenet. The model has 54,361,668 parameters and requires 217.4MB of storage space. We use a batch size of 24 images, a momentum of 0.9, and weight decay of 0.9. The learning rate is set to 4×10^{-3} with no adjustments until the end of the training.

In the validation set, the best SSD model provides a mAP=73.6 at iteration 72,325. Over the test set, that same model reaches a mAP=87.9, which is surprisingly the best mAP value of all experiments.

C. YOLO Full

We train a YOLO network [36] using the *Darknet-19* [12] convolutional architecture that comprises 19 convolutional layers. Such a model was trained as described in [36] starting from an Imagenet [42] pre-trained model and fine-tuned over PTLD. The model comprises a segment with convolutional layers and residual connections in a total of $50,552,672$ parameters. Each model requires 202.4MB of storage size. We use a batch of 64 images, a momentum of 0.9, and weight decay of 5×10^{-4}. The learning rate schedule follows the original work's recommendation: it starts with 10^{-3}, and then goes to 10^{-4} after $40,000$ iterations, and to 10^{-5} after $60,000$. The training stops after $100,000$ iterations. Each model took 10 hours to be trained.

The model that performs best on validation data was observed at iteration $30,000$, resulting in mAP=74.4. In the test set, this model provides a mAP=76.1.

D. YOLO Tiny

We also trained a faster YOLO alternative, namely YOLO Tiny. To speedup the process, YOLO Tiny employs a simplified convolutional architecture that comprises only a portion of the *Darknet-19* [12] resources: 9 convolutional layers, resulting in $15,773,616$ parameters. Each model requires only 63.1MB of storage size. The network was trained as described in [36], fine-tuning an Imagenet [42] pre-trained model over PTLD. We also use the validation set to observe the training evolution. We use a batch size of 64 images, a momentum of 0.9, and weight decay of 5×10^{-4}. The learning rate schedule is the same as YOLO's training: starts with 10^{-3}, going to 10^{-4} after $40,000$ iterations, and to 10^{-5} after $60,000$ iterations. We stopped training after $100,000$ iterations. Each model took ≈ 5 hours to be trained.

The best model on the validation set was observed at iteration $40,000$, providing a mAP=44.6. Such a model provides a mAP=35.8 in PTLD test set.

E. Discussion

By creating PTLD, our main goal is to assist people with visual impairment crossing streets that have pedestrian traffic lights (PTL), informing if there is a PTL and whether it is allowing or prohibiting people to cross the street. Therefore, a first empirical evaluation with state-of-the-art methods for object detection over PTLD is essential for reaching that goal.

Our evaluation comprises 3 different methods with state-of-the-art results in the most well-known datasets for object detection. We established a time frame of 96 hours (4 days) of training for all methods, proceeding the training in the same hardware and software environment.

By looking at the results presented in Table III, SSD provides the best mAP and AP for all classes. The Faster R-CNN presents results inferior to SSD and YOLO, which is not expected given the robustness of this method. Notwithstanding, we can see in Table II that the best model of Faster R-CNN was achieved in the last iteration, which indicates that we would probably achieve better results if it was allowed to train for a longer period.

When adopting a *Darknet-19* architecture, YOLO reaches results inferior than Faster R-CNN and SSD, but still promising for being embedded into a smartphone considering its prediction speed shown in Table IV The simplest model, YOLO Tiny, produces the worst results of all methods, as one could have previously expected, though with the fastest prediction times.

TABLE III
PTLD TEST DETECTION RESULTS. MAP AND AP RESULTS.

Method	mAP	go	stop	off
Faster R-CNN	77.8	86.5	83.4	63.7
SSD	**87.9**	**90.4**	**93.4**	**79.9**
YOLO Full	76.2	74.6	90.4	63.5
YOLO Tiny	35.8	43.2	57.9	64.0

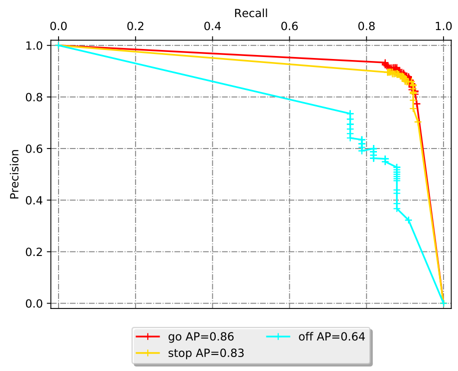
Figure 5 illustrates the performance of all models contrasting precision and recall, while the threshold of confidence varies from 0 to 1 by increments of 0.01. We assume predictions as correct when the IOU with the ground truth was greater than 0.5. The overall performance of each method can be synthesized by mAP and per-class AP. Note that YOLO presents dispersal PR points (Figure 5c), especially when compared to SSD (Figure 5b), which in turn presents a dense point distribution. This is due to a significant concentration of the confidence score for the correct predictions of SSD.

With respect to prediction times, Table IV confirms our intuition that YOLO Tiny reaches the fastest predictions. YOLO Full also presents impressive results with FPS=100, which allows its application in video-based applications even considering per-frame approaches. SSD's prediction speed is also quite fast. YOLO Full has a speedup of $2\times$ over SSD, and YOLO Tiny presents a speedup of $1.5\times$ over YOLO Full.

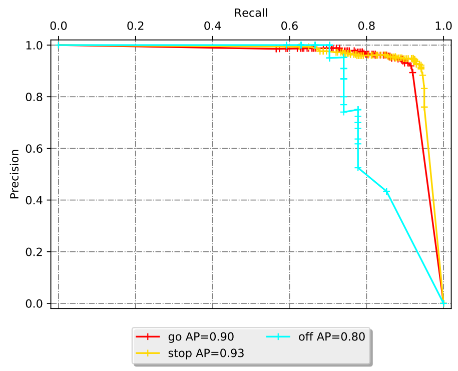
TABLE IV
PREDICTION TIME PER IMAGE (IN SECONDS).

Method	Prediction speed (in seconds)	FPS
Faster R-CNN	2.700 ± 0.330	0.37
SSD	0.020 ± 0.004	50
YOLO Full	0.010 ± 0.000	100
YOLO Tiny	$\mathbf{0.004} \pm 0.000$	**250**

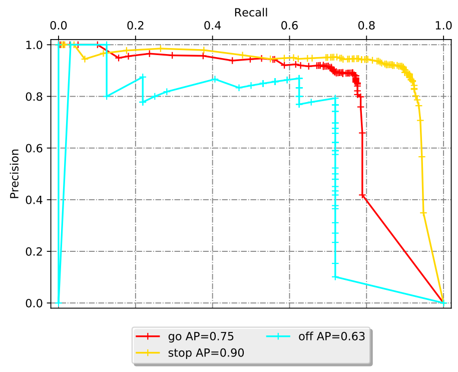
To illustrate qualitative results obtained with the set of experiments that were performed in this study, Figure 6 presents 3 randomly selected samples disposed in 4 rows, each corresponding to one of the object detection methods. The samples are part of our test set and show the contrast of the ground truth (yellow) and the predictions (red). For reference, the ground truth boxes were plotted in yellow and are overlapped by the predictions. Due to the divergent distribution of confidence scores presented by each method (see Figure 5), we tune the confidence threshold θ for each



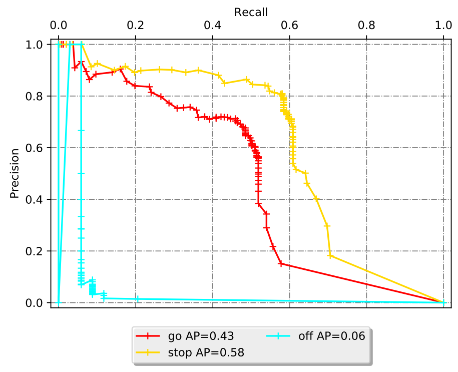
(a) Faster R-CNN



(b) SSD



(c) YOLO



(d) YOLO Tiny

Fig. 5. PR curves for each object detection method trained on PTLD.

model according to the observations in the validation set. The values that are used are $\theta = [0.34, 0.02, 0.17, 0.28]$ for Faster R-CNN, SSD, YOLO, and YOLO Tiny, respectively.

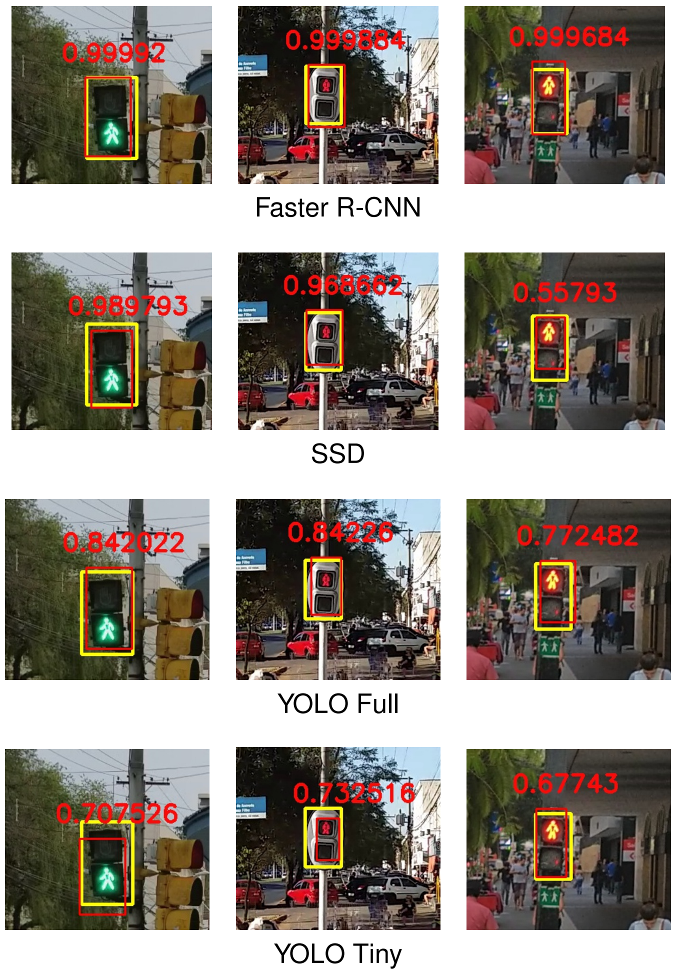


Fig. 6. Model generalization evaluation.

By looking at the predictions in Figure 6, note that they are all quite similar. We can see that there are marginal differences in the positions of the boxes. Regarding the confidence, we can see that YOLO Tiny is the one with the greatest discrepancy.

F. Tests with other datasets

To evaluate the generalization capability of our approach, we decided to test it over the German dataset [8], [11]. We can see in Figure7 that the German standard for pedestrian traffic lights is substantially different from the Brazilian standard (Figure 1), mainly regarding the 3-part shape of the lights.

The German dataset was used in two studies [8], [11], and it was analyzed using the following measures: Recall, $R = TP/(TP + FN)$; and Precision, $P = TP/(TP + FP)$. Note that, in those papers, there is no analysis regarding intersection over union (IOU). Predictions are correct simply if the predicted box lies anywhere over the ground-truth box. For that reason, in this comparison we use the same approach for computing true (false) positives (negatives).

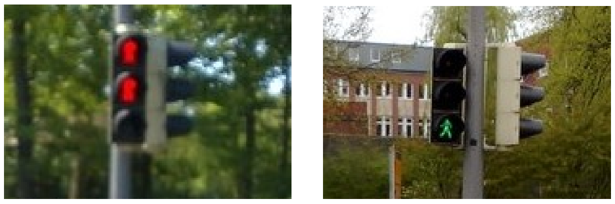


Fig. 7. German standard pedestrian traffic lights samples.

TABLE VI
COMPARISON OF OUR BEST MODEL WITH THE WORK BY ROTERS ET AL. [8] AND CHENG ET AL. [11] IN THE GERMAN DATASET.

Class	Measure	Faster R-CNN	Roters [8]	Cheng [11]
Go	Recall (%)	**97.7**	55.3	57.3
	Precision (%)	93.5	**100**	97.6
Stop	Recall (%)	**98.9**	52.4	90.3
	Precision (%)	90.2	**100**	98.3

Instead of re-training all methods, we decided to test all models previously trained on the PTLD dataset over the German data, and then choose only the model with the best result to perform fine-tuning that dataset. It is important to mention that the images of that dataset were not included in our own dataset and, therefore, they were not used at all when training our original models.

Table V shows a comparison of all our methods when trained in PTLD and tested over the German data. For this dataset, Faster R-CNN presented the best results for both classes *Go* and *Stop*. The other methods presented quite low recall values, and for this reason they were not fine-tuned over the German data.

TABLE V
GENERALIZATION ABILITY OF OUR MODELS IN THE GERMAN DATASET.

Class	Measure	Faster	SSD	YOLO Full	YOLO Tiny
Go	Recall (%)	**50.0**	21.3	39.8	29.1
	Precision (%)	98.4	**100**	**100**	98.6
Stop	Recall (%)	**85.4**	60.6	65.8	24.8
	Precision (%)	98.6	99.2	98.9	**100**

Now that we have selected the best model (Faster R-CNN), we fine-tuned it over the German training data. This dataset has only two classes, Go and Stop, so the number of classes has changed to 2 and the batch size is set to 1 image. Momentum is set to 0.9 and the learning rate is kept fixed in 2×10^{-4}. According to the validation data, the best model was created at iteration $200,000$, which provides a mAP=90.3. After evaluating such a model in the test set, the resulting mAP was 75.2. It reached $200,000$ iterations in 1 day of training time.

Comparing the results of our model with the results presented in the work by Roters et al. [8] and Cheng et al. [11], note that our approach presents a substantially superior recall score for both classes, which shows the high detection rate of our model. The precision score is smaller but competitive, probably due to the fact that our model performs much more detections than the previous state-of-the-art, and hence it ends up detecting false positives.

Therefore, even though SSD seemed to be a better choice in the tests with our dataset (Table III), Faster R-CNN seems to be more promising, especially considering it was still learning when we stopped training. The generalization of the models in the German dataset clearly indicate that Faster R-CNN seems to be a better choice. The obvious limitation is that Faster R-CNN is much more time-consuming during prediction time, so a sampling strategy needs to be employed in order to make it run in real-time for helping visually-impaired people.

VI. CONCLUSIONS

Crossing streets autonomously is still a challenge for visually-impaired people. Hence, we propose an approach capable of detecting pedestrian traffic lights and their current state based on images acquired by mobile devices via state-of-the-art computer vision techniques that rely on deep neural networks. We show that deep learning approaches can achieve good results considering all challenges involved in the recognition of a pedestrian traffic light across the street in an uncontrolled environment.

Our contributions are mainly regarding the novel public dataset with 4,399 labeled images of pedestrian traffic lights and the controlled comparison among the deep learning state-of-the-art methods for classification and detection. Another contribution is that our approach, using our pre-trained models, can easily be used in other datasets and obtain state-of-the-art results. We also show the feasibility of our approach to be embedded in a mobile device. We are now working to embed our models in a mobile device and develop the app for helping the visually impaired. Thus, for future work we intend to perform tests of the app with users that present visual impairment, and make it available for all interested audiences.

ACKNOWLEDGMENT

We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research. The authors would also like to thank all the graduate and undergraduate students and family members who helped to generate the dataset.

REFERENCES

[1] V. Murali and J. Coughlan, "Smartphone-based crosswalk detection and localization for visually impaired pedestrians," in *IEEE International Conference on Multimedia and Expo Workshops (ICMEW)*, July 2013, pp. 1–7.
[2] X. Li, H. Ma, X. Wang, and X. Zhang, "Traffic light recognition for complex scene with fusion detections," *IEEE Transactions on Intelligent Transportation Systems*, vol. PP, no. 99, pp. 1–10, 2017.
[3] W. Liu, S. Li, J. Lv, B. Yu, T. Zhou, H. Yuan, and H. Zhao, "Real-time traffic light recognition based on smartphone platforms," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 5, pp. 1118–1131, May 2017.

[4] M. B. Jensen, M. P. Philipsen, A. Møgelmose, T. B. Moeslund, and M. M. Trivedi, "Vision for looking at traffic lights: Issues, survey, and perspectives," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 7, pp. 1800–1815, July 2016.

[5] K. Behrendt, L. Novak, and R. Botros, "A deep learning approach to traffic lights: Detection, tracking, and classification," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 1370–1377.

[6] M. B. Jensen, K. Nasrollahi, and T. B. Moeslund, "Evaluating state-of-the-art object detector on challenging traffic light data," in *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, July 2017, pp. 882–888.

[7] G. G. Lee and B. K. Park, "Traffic light recognition using deep neural networks," in *2017 IEEE International Conference on Consumer Electronics (ICCE)*, Jan 2017, pp. 277–278.

[8] J. Roters, X. Jiang, and K. Rothaus, "Recognition of traffic lights in live video streams on mobile devices," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 21, no. 10, pp. 1497–1511, Oct 2011.

[9] Y. K. Kim, K. W. Kim, and X. Yang, "Real time traffic light recognition system for color vision deficiencies," in *2007 International Conference on Mechatronics and Automation*, Aug 2007, pp. 76–81.

[10] J. Balcerek, K. Piniarski, M. Urbanek, K. Szałecki, and A. Konieczka, "Mobile applications for driver and pedestrian assistance," in *2015 Signal Processing: Algorithms, Architectures, Arrangements, and Applications (SPA)*, Sept 2015, pp. 197–202.

[11] R. Cheng, K. Wang, K. Yang, N. Long, J. Bai, and D. Liu, "Real-time pedestrian crossing lights detection algorithm for the visually impaired," *Multimedia Tools and Applications*, Dec 2017. [Online]. Available: https://doi.org/10.1007/s11042-017-5472-5

[12] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," *arXiv preprint arXiv:1612.08242*, 2016.

[13] 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.

[14] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European conference on computer vision ECCV*. Springer, 2016, pp. 21–37.

[15] J. Ying, J. Tian, and L. Lei, "Traffic light detection based on similar shapes searching for visually impaired person," in *2015 Sixth International Conference on Intelligent Control and Information Processing (ICICIP)*, Nov 2015, pp. 376–380.

[16] X. Kou, Y. Wei, and M. Lee, "Vision based guide-dog robot system for visually impaired in urban system," in *2013 13th International Conference on Control, Automation and Systems (ICCAS 2013)*, Oct 2013, pp. 130–135.

[17] S. Alghamdi, R. van Schyndel, and I. Khalil, "Safe trajectory estimation at a pedestrian crossing to assist visually impaired people," in *2012 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, Aug 2012, pp. 5114–5117.

[18] M. P. Philipsen, M. B. Jensen, A. Møgelmose, T. B. Moeslund, and M. M. Trivedi, "Traffic light detection: A learning algorithm and evaluations on challenging dataset," in *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, Sept 2015, pp. 2341–2345.

[19] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: a new learning scheme of feedforward neural networks," in *2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No.04CH37541)*, vol. 2, July 2004, pp. 985–990 vol.2.

[20] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, June 2005, pp. 886–893 vol. 1.

[21] A. Dixit and N. P. Hegde, "Image texture analysis - survey," in *2013 Third International Conference on Advanced Computing and Communication Technologies (ACCT)*, April 2013, pp. 69–76.

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

[23] Y. Freund and R. E. Schapire, "A short introduction to boosting," in *In Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*. Morgan Kaufmann, 1999, pp. 1401–1406.

[24] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," *Computer vision–ECCV 2006*, pp. 404–417, 2006.

[25] S. Mascetti, D. Ahmetovic, A. Gerino, C. Bernareggi, M. Busso, and A. Rizzi, "Robust traffic lights detection on mobile devices for pedestrians with visual impairment," *Computer Vision and Image Understanding*, vol. 148, pp. 123 – 135, 2016, special issue on Assistive Computer Vision and Robotics.

[26] ——, *Supporting Pedestrians with Visual Impairment During Road Crossing: A Mobile Application for Traffic Lights Detection*. Cham: Springer International Publishing, 2016, pp. 198–201. [Online]. Available: https://doi.org/10.1007/978-3-319-41267-2_27

[27] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, vol. 1, 2001, pp. I–511–I–518 vol.1.

[28] Y. LeCun, L. Jackel, B. Boser, J. Denker, H. Graf, I. Guyon, D. Henderson, R. Howard, and W. Hubbard, "Handwritten digit recognition: Applications of neural network chips and automatic learning," *IEEE Communications Magazine*, vol. 27, no. 11, pp. 41–46, 1989.

[29] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012, pp. 1097–1105.

[30] L. Chen, S. Wang, W. Fan, J. Sun, and S. Naoi, "Beyond human recognition: A CNN-based framework for handwritten character recognition," in *Asian Conference on Pattern Recognition*, 2015.

[31] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[32] G. S. Simões, J. Wehrmann, R. C. Barros, and D. D. Ruiz, "Movie genre classification with convolutional neural networks," in *International Joint Conference on Neural Networks*, 2016, pp. 259–266.

[33] J. Wehrmann, R. C. Barros, G. S. Simões, T. S. Paula, and D. D. Ruiz, "(Deep) Learning from Frames," in *BRACIS*, 2016, pp. 1–6.

[34] J. Wehrmann and R. C. Barros, "Convolutions through time for multi-label movie genre classification," in *Proceedings of the Symposium on Applied Computing*. ACM, 2017, pp. 114–119.

[35] ——, "Movie genre classification: A multi-label approach based on convolutions through time," *Applied Soft Computing*, vol. in press, 2017.

[36] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *IEEE CVPR*, 2016, pp. 779–788.

[37] J. Monteiro, R. Granada, R. C. Barros, and F. Meneguzzi, "Deep neural networks for kitchen activity recognition," in *International Joint Conference on Neural Networks*, 2017, pp. 2048–2055.

[38] J. Monteiro, J. P. Aires, R. Granada, R. C. Barros, and F. Meneguzzi, "Virtual guide dog: An application to support visually-impaired people through deep convolutional neural networks," in *International Joint Conference on Neural Networks*, 2017.

[39] J. Wehrmann, A. Mattjie, and R. C. Barros, "Order embeddings and character-level convolutions for multimodal alignment," *CoRR*, vol. abs/1706.00999, 2017. [Online]. Available: http://arxiv.org/abs/1706.00999

[40] W. Becker, J. Wehrmann, H. E. Cagnini, and R. C. Barros, "An efficient deep neural architecture for multilingual sentiment analysis in twitter," in *International FLAIRS Conference*, 2017.

[41] J. Wehrmann, W. Becker, H. E. Cagnini, and R. C. Barros, "A character-based convolutional neural network for language-agnostic twitter sentiment analysis," in *International Joint Conference on Neural Networks*, 2017, pp. 2384–2391.

[42] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[43] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes challenge: A retrospective," *International Journal of Computer Vision*, vol. 111, no. 1, pp. 98–136, Jan. 2015.

[44] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. Zitnick, "Microsoft COCO: Common objects in context," in *European Conference on Computer Vision*, 2014, pp. 740–755.

[45] R. Girshick, "Fast r-cnn," in *International Conference on Computer Vision (ICCV)*, 2015.

[46] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning." in *AAAI*, 2017, pp. 4278–4284.