Traffic Signs Detection and Recognition System in Snowy Environment Using Deep Learning

Hamou Chehri[1], Abdellah Chehri[2] , Rachid Saadane[3]

[1] Bell-Canada, 671 Rue de la Gauchetière Ouest, Montréal, Québec, H3B 2M8, Canada.
[2] Department of Applied Sciences, University of Québec in Chicoutimi, Canada
[3] SIRC/LaGeS-EHTP, EHTP Km 7 Route El Jadida, Oasis, Morocco.
hamou.chehri@uqat.ca; achehri@uqac.ca; saadane@ehtp.ac.ma

Abstract. A fully autonomous car does not yet exist. But the vehicles have continued to gain in range in recent years. The main reason? The dazzling progress made in artificial intelligence, in particular by specific algorithms, known as machine learning. These example-based machine learning methods are used in particular for recognizing objects in photos. The algorithms developed for the detection and identification must respond robustly to the various disturbances observed and take into account the variability in the signs' appearance. Variations in illumination generate changes in apparent color, shadows, reflections, or backlighting. Besides, geometric distortions or rotations may appear depending on the viewing angle and the panels' scale. Their appearance may also vary depending on their state of wear and possible dirt, damage. In this work, to improve the accuracy of detection and classification of sign road partially covered by snow, we use the Fast Region-based Convolutional Network method (Fast R-CNN) model. To train the detection model, we collect an image dataset composed of multi-class of road signs. Our model can simultaneously multi-class of a road sign in nearly real-time.

Keywords: Deep learning; Automatic Classification; traffic sign; detection.

1 Introduction

The detection and recognition of road signs is a major issue in the analysis of road scenes by image processing. There are many applications, such as route calculation with estimated travel times, the development of tools for the management and maintenance of road assets, real-time driving assistance systems or, in connection with robotics, vehicle automation or even the development of a new generation of multimedia tools on the web for geographic 3D navigation.

Whatever the applications, detection and recognition methods come up against the difficulties linked both to the uncontrolled nature of the shots used and to the variability in appearance of the objects sought. Signs are manufactured and standardized objects whose shape, dimensions, color and position are fixed by standards. variations in illumination generate changes in apparent color, shadows, reflections or backlighting. In addition, geometric distortions or rotations may appear depending on the viewing angle and the scale of the panels. Their appearance may also vary depending on their state of wear and possible dirt, damage or partially covered by snow.

The algorithms developed for the detection and identification of road signs must respond in a robust manner to the various disturbances observed and take into account the variability in appearance of the signs. These algorithms used in general the methods by image processing consist of two steps: the detection of signs in the road scene and the recognition of their type. These algorithms can be roughly classified into three categories: Color-based, Shape-based, and Machine Learning-based methods.

The Color-based methods, segmentation is applied to detect regions of interest. There are specific characteristic colors of traffic signs: red, blue, and yellow. These characteristics, however, indicate sensitivity to various factors, such as the age of signs and the variation of light, which make the segmentation an arduous process. In order to overcome this problem, authors are working on different color spaces [1]-[3].

The Shape-based methods, in this approach, do not absolutely consider color segmentation as a discriminative feature due to its sensitivity to various factors such as the distance of the target, weather conditions, time of the day, and reflection of the signs. Conversely, detection of the signs is made from the edges of the image analyzed by structural or comprehensive approaches. Shape-based methods are generally robust than colorimetric methods by reason they can process images in grayscale and treat their gradients. However, they are costly in computation time, given the fact that the rate of treatment depends largely on the number of detected edges. However, shape-based methods can treat grayscale images; in some countries, such as Japan, there are pairs of different signs in the highway code which, when converted to grayscale, appear exactly the same. To be able to distinguish them, an amount of color information is absolutely needed [4]. On the other hand, some authors adopt the color feature to localize the region of interest and complete with shape methods in order to detect the signs position and recognize its geometric form.

Both Color-based and Shape-based have common weakness in several factors such as lighting changes, occlusions, scale change, rotation, and translations. However, these problems could also be treated using Machine Learning-based methods to learn these features to realize surface detection, but it requires a large database of annotated data.

The CNN-based deep learning for surface detection is that CNN can simultaneously achieve the automatic extraction and recognition of elements in a network, and get rid of the trouble of manually extracting features [5]. The CNN-based deep learning makes detection more accurate, which provides a novel deep learning approach in the industrial and research field [6].

With the rapid development of computer technology, machine vision has been widely applied in object detection and classification. In recent years, machine learning has driven advances in many different fields [7]. We attribute this success to the invention of more sophisticated machine learning models [8], the availability of large datasets for tackling problems in these fields [9], and the development of software platforms that enable the easy use of large amounts of computational resources for training such models on these large datasets [10].

Detection and recognition of traffic-sign in snowy environment remains an open question. Various previous works have addressed the traffic-sign recognition and detection, However, several of them focused only on traffic-sign detection in clear visibility.

This paper attempts to train its convolutional neural network for partially covered traffic signs. First, a complete dataset of road signs with 20 classes. Next, the sample images were preprocessed through faster RCNN to realize the intelligent detection of the object. Finally, we will have a program that can identify and draw boxes around specific objects in pictures, videos, or in a webcam feed.

This paper is organized as follows. The related works on road signs detection and classification is given in section 2. In Section 3, we give an overview of the object detection techniques. The sampling and image preprocessing procedures are given in Section 4. Section 5 concludes the paper.

2 Related works

Since 1989 [11] researchers started to apply machine learning methods for sign detection. Subject of machine learning is to study how to use computer imitate human learning activities, and to discover self-improvement methods of computers that to gather new skills and new knowledge, classify existing knowledge, and deliberately improve the performance [12].

Over the recent few years, deep learning has had unprecedented success in fields like speech recognition, image classification, etc. Deep learning is a subset of machine learning in Artificial Intelligence (AI) that has networks which are capable of learning unsupervised from data that is unstructured or unlabeled and is about learning multiple levels of representation and abstraction that help to make sense of data such as images, sound, and text [13]. So, it is making them very effective for traffic sign detection. Especially when track any objects in an actual video sequence where these objects are non-rigid, the background of the scene is not fixed and in the case of several objects in the same scene.

Due to the success of CNN in traffic sign classification, the authors in [14] propose a lightweight and optimized ConvNet with sliding window to detect traffic signs in high-resolution images. The accuracy rate detection achieved is 99.89% on the German Traffic Sign Detection Benchmark dataset. Time execution on GPU (GeForce GTX 980) is 26.506ms which is equal to processing 3772 frames per second. Obtained results make this approach a real-time application.

Wu et al. [15] use convolutional neural networks CNN to localize and recognize traffic sign, firstly they use support vector machine to transform the original image from RGB to grayscale to avoid falling into the problem of sensitivity to color difference due to various lighting conditions, secondly they use the fixed layer in the CNNto localize region of interest which are similar to traffic sign, and the learnable layers are used to extract discriminant features for classification. They use GTSDB as a dataset and they obtained 99.73% in warning signs and 97.62% in mandatory signs, but it is too far from a real-time application. Cascaded convolutional neural networks (CNNs) are recently used by [16] to reduce false positive regions detected using the local binary pattern (LBP) feature detector combined with the AdaBoost classifier.

Xiong et al. [17] trained a traffic sign detection model based on deep CNNs using Region Proposal Network (RPN) in Fast R-CNN. Running on the hardware environment of NVIDIA GTX980Ti 6GB GPU, the average detection time is about 51.5 ms per image with the detection rate above 99% in continuous image sequence. The database they used is Chinese traffic sign with 7 main categories.

Another Faster R-CNN-based model was proposed in [18], with two parts of using selective search to detect candidate regions firstly and then using CNNs to extract features, make classifications and modify parameters.

New way to detect traffic signs was discovered by Peng et al. [19]. They use Faster R-CNN based on Region Proposal Networks to achieve 90% of accuracy with NVIDIA GTX 1070 8 GB GPU, Intel Core i5, 16 GB RAM on GTSDB dataset.

3 Machine Learning-based methods

The purpose of using object detection algorithms is to draw a bounding box around the object of interest to locate it within the image. You might not necessarily bring just one bounding box in an object detection case. There could be many bounding boxes representing different objects of interest within the image, and you would not know how many beforehand. Therefore, an algorithm like Faster R-CNN has been developed to find these occurrences and find them fast.

3.1 R-CNN Algorithm

R-CNN Algorithm has been developed to bypass the problem of selecting a considerable number of regions, Ross et al. [20] proposed a method where we use selective search to extract just 2000 regions from the image, and he called them region proposals. Therefore, instead of trying to classify a considerable number of regions, you can work with 2000 regions. These 2000 candidate region proposals are warped into a square and fed into a convolutional neural network that produces a 4096-dimensional feature vector as output (see **Error! Reference source not found.**).

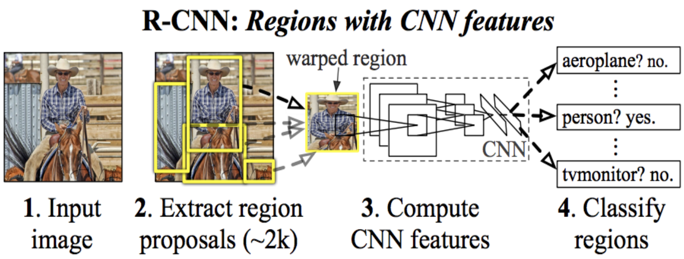


Fig. 1. A figure caption is always placed below the illustration. Short captions are centered, while long ones are justified. The macro button chooses the correct format automatically [21].

3.2 Fast R-CNN

The same author of the previous paper (R-CNN) solved some of R-CNN's drawbacks to building a faster object detection algorithm, and it was called Fast R-CNN [19]. The approach is similar to the R-CNN algorithm (see **Error! Reference source not found.**). Instead of feeding the region proposals to the CNN, we supply the input image to the CNN to generate a convolutional feature map. The reason "Fast R-CNN" is faster than R-CNN is because you don't have to feed 2000 region proposals to the convolutional neural network every time. Instead, the convolution operation is done only once per image, and a feature map is generated from it [20] – [21].

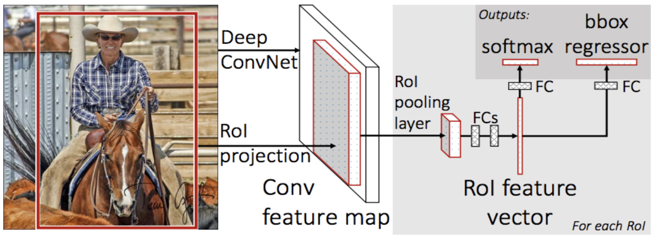


Fig. 2. Fast R-CNN [21].

3.3 Faster R-CNN

Both of the above algorithms (R-CNN & Fast R-CNN) use selective searches to determine the region proposals. Selective search is a slow and time-consuming process affecting the performance of the network. Therefore, Ren et al. [22] came up with an object detection algorithm that eliminates the selective search algorithm and lets the network learn the region proposals (see **Error! Reference source not found.**).

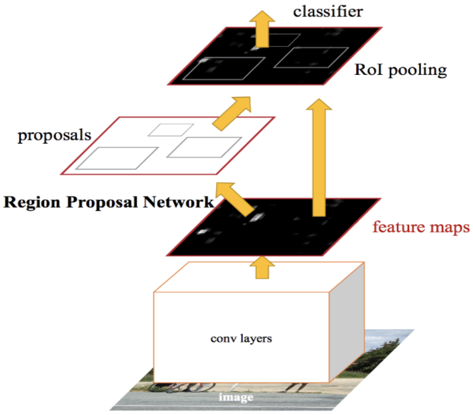


Fig. 3. Faster R-CNN [21].

Similar to Fast R-CNN, the image is provided as an input to a convolutional network which provides a convolutional feature map. Instead of using selective search algorithm on the feature map to identify the region proposals, a separate network is used to predict the region proposals. The predicted region proposals are then reshaped using a RoI pooling layer which is then used to classify the image within the proposed region and predict the offset values for the bounding boxes.

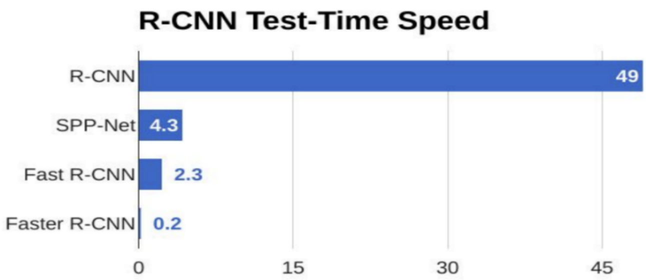


Fig. 4. Fast Comparison of test-time speed of object detection algorithms [20].

From **Error! Reference source not found.**, you can see that Faster R-CNN is much faster than its predecessors. Therefore, it can even be used for real-time object detection.

Tensorflow detection model zoo provides a collection of detection models pre-trained. Some model has high speed with lower accuracy, other models such as Faster R-CNN have a lower speed but a higher efficiency.

4 Sampling and Image Preprocessing

4.1 Experimental Environment

Software environment: Windows 10 64-bit operating system, TensorFlow 1.15.1, Python 3.8.0 64-bit.

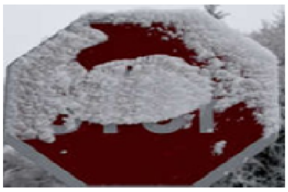
Hardware environment: Intel (R) Core (TM) i7-7700 CPU@3.80GHz processor, 16.00 GB memory, NVIDIA GeForce GTX 1050, 512 SSD hard disk.

4.2 Data Collection

General image datasets such as ImageNet [23] and Microsoft COCO [24] have been generated by downloading Internet images retrieved by search engines using keywords.

To mimic a real-world application scenario, we select the images for sign road partially covered by snow [24]-[25].

Traffic signs in Canada follow international patterns, and can be classified into three categories: warnings (mostly yellow rectangles with a black boundary and information), prohibitions (mostly white surrounded by a red circle and also possibly having a diagonal bar), and mandatory (mostly green circles with white information). Other signs exist that resemble traffic-signs but are in fact not; like the one illustrated in **Fig. 5.** (d). Such signs are placed in an ‘other’ class of a particular category. The data samples were composite of a total of 600 images, 150 images for each class. The dataset contains the pictures of 4 classes of the road signs.



(a) (b)

(c) (d)

Fig. 5. The different classes one our data set.

4.3 Sampling

All implementation in this part has been done using an OpenCV environment. Pre-implemented functionality from the OpenCV library has been used to keep the program robust and will be stated when presented. This section will show a step-by-step solution of the image processing from source images to extracted features data of each leaf.

From the data collection images, 80% were randomly selected and allocated to the training set, and the remaining 20% were allocated into the test set.

4.4 Labeling

The images collected were next annotated by hand. We used the LabelImg tool for labeling desired objects in every picture. Then draw a box around each object in each image. LabelImg saves a .xml file. This file will contain the label data for each image. These .xml files will be used to generate TFRecords, which are one of the inputs to the TensorFlow trainer.

4.5 Generate training Data

After labeling for all objects, TFRecords were generated, serving as input data to the TensorFlow training model. The image .xml data were used to create .csv files containing all the data for the train and test images. This creates a train_labels.csv and test_labels.csv file in the training folder.

4.6 Create label map and configure training

The last thing to do before training is to create a label map and edit the training configuration file. The label map tells the trainer what each object is by defining class names' mapping to class ID numbers. The label map ID numbers should be the same as what is defined in the generate_tfrecord.py file.

The fault detection for traffic road must be configured. It defines which model and what parameters will be used for training. There are several changes to make to the .config file, mainly changing the number of classes and examples, and adding the file paths to the training data.

Fig. 6. Example of labeling traffic sign using LabelImg tool.

4.7 Run and time the data training

Each step of training reports the loss. It will start high and get lower and lower as training progresses. For our training on the Faster-RCNN-Inception-V2 model, it started over1 and dropped below 0.5 after 2200 steps. The model was left to train until the loss consistently drops below 0.10, which will take about 5150 steps (Fig. 7).

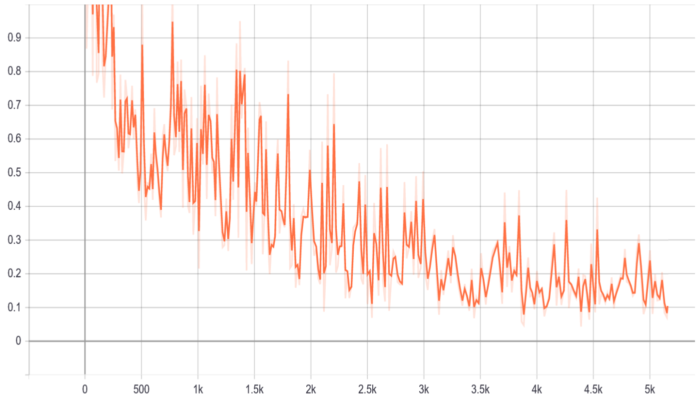


Fig. 7. One important graph is the Loss graph, which shows the overall loss of the classifier over time.

5 Conclusion

The system architecture has proved to be a promising approach at this stage of development. All though there is still major parts of the system to be developed, its modularity makes it easy to develop and understand.

The implemented program works well and the training mode is fully functional. However, a classifier to be used in the spraying mode in the classification part of the system, is to be implemented. In this paper, the use of Python has been applied to examine the different classifiers. This is done in order to investigate the results and the performance before choosing a classifier and implementing it into the program. On the other hand, a framework for the program has been created and is easy to develop further.

References

1. De La Escalera A, Moreno LE, Salichs MA, Armingol JM (1997) Road traffic sign detection and classification. IEEE Trans Indus Electron 44(6):848–859
2. Yakimov P (2015) Traffic signs detection using tracking with prediction. In: International conference on E-business and telecommunications Colmar, France, Springer, pp 454–467
3. De la Escalera A, Armingol JM,MataM(2003) Traffic sign recognition and analysis for intelligent vehicles. Image Vis Comput 21(3):247–258
4. Ruta A, Li Y, Liu X (2010) Real-time traffic sign recognition from video by class-specific discriminative features. Pattern Recogn 43(1):416–430
5. T. Mikolov, K. Chen, G. Corrado, and J. Dean. Ef- ficient estimation of word representations in vector space. In Proceedings of ICLR Workshops Track, 2013.
6. A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In Proceedings of NIPS, pages 1106–1114, 2012.
7. O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei- Fei. ImageNet Large Scale Visual Recognition Chal- lenge. International Journal of Computer Vision, 115(3):211–252, 2015.
8. J. Dean, G. S. Corrado, R. Monga, K. Chen, M. Devin, Q. V. Le, M. Z. Mao, M. Ran- zato, A. Senior, P. Tucker, K. Yang, and A. Y. Ng. Large scale distributed deep net- works. In Proceedings of NIPS, pages 1232–1240, 2012.
9. Martín Abadi, et al. 2016. TensorFlow: A system for large-scale machine learning. In Proceedings of the USENIX Symposium on Operating Systems Design and Implementation.
10. Molenaar D., Sadler B.A. (2014) Anode Rodding Basics. In: Grandfield J. (eds) Light Metals 2014. Springer.
11. Lubin, Jones and Kornhauser, "Using back-propagation networks to assess several image representation schemes for object recognition," International Joint Conference on Neural Networks, 1989.
12. H. Wang, et al. "A Brief Review of Machine Learning and Its Application," 2009 International Conference on Information Engineering and Computer Science, 2009.
13. Investopedia, "definition of 'Deep Learning'," [Online]. Available: http://www.investopedia.com/terms/d/deep-learning.asp.
14. Aghdam H. H, Heravi EJ, Puig D (2016) A practical approach for detection and classification of traffic signs using Convolutional Neural Networks. Robot Auton Syst 84:97–112
15. Wu Y, Liu Y, Li J, Liu H, Hu X (2013) Traffic sign detection based on convolutional neural networks. In: The 2013 international joint conference on neural networks (IJCNN), pp 1–7.

16. Zang D, Zhang J, Zhang D, BaoM, Cheng J, Tang K (2016) Traffic sign detection based on cascaded convolutional neural networks. In: 17th IEEE/ACIS international conference on software engineering, artificial intelligence, networking and parallel/distributed computing (SNPD), 2016, IEEE, pp 201–206
17. C. Xiong, C. Wang, W. Ma and Y. Shan, "A traffic sign detection algorithm based on deep convolutional neural network," IEEE ICSIP, 2016.
18. Z. Zuo, K. Yu, Q. Zhou, X. Wang and T. Li, "Traffic Signs Detection Based on Faster R-CNN," 2017 IEEE 37th International Conference on Distributed Computing Systems Workshops, 2017.
19. E. Peng, F. Chen and X. Song, " Traffic Sign Detection with Convolutional Neural Networks ", ICCSIP in, Singapore, 2017.
20. Rohith Gandhi, R-CNN, Fast R-CNN, Faster R-CNN, YOLO — Object Detection Algorithms Understanding object detection algorithms, 2018.
21. S. Ren, K. He, R. Girshick and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks", 28th Int. Conf. Neural Inf. Process. Syst., pp. 91-99, 2015.
22. J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hiera chical Image Database. In CVPR09, 2009.
23. T. Lin, M. Maire, S. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollar, and C. L. Zitnick. Microsoft COCO: common objects in context. CoRR, abs/1405.0312, 2014.
24. Chehri A., Fortier P. (2019) Low-Cost Localization and Tracking System with Wireless Sensor Networks in Snowy Environments. In: Chen YW., Zimmermann A., Howlett R., Jain L. (eds) Innovation in Medicine and Healthcare Systems, and Multimedia. Smart Innovation, Systems and Technologies, vol 145. Springer, Singapore.
25. Chehri A., Fortier P. Wireless Positioning and Tracking for Internet of Things in Heavy Snow Regions. In: Zimmermann A., Howlett R., Jain L. (eds) Human Centred Intelligent Systems. Smart Innovation, Systems and Technologies, vol 189. Springer, Singapore, June 2020.