Computer Vision System for Speed Limit Traffic Sign Recognition

Adelina D. Salimullina
Higher School of Electronics and Micro-Electro-Mechanical Systems
Peter the Great St. Petersburg Polytechnic University
Saint Petersburg, Russia
salimullina.ad@edu.spbstu.ru

Dmitry O. Budanov
Higher School of Electronics and Micro-Electro-Mechanical Systems
Peter the Great St. Petersburg Polytechnic University
Saint Petersburg, Russia
budanov_do@spbstu.ru

2022 Conference of Russian Young Researchers in Electrical and Electronic Engineering (ElConRus) | 978-1-6654-0993-3/22/$31.00 ©2022 IEEE | DOI: 10.1109/ElConRus54750.2022.9755744

***Abstract*—The paper is devoted to the development of a portable speed limit traffic sign recognition system widely available for general use. A shape-based approach has been applied to detect a speed limit sign. The sign recognition has been carried out with a convolutional neural network. A single board computer was used as a prototype of a hardware part of the proposed system. The hardware and software parts of the proposed system have been developed and tested. The traffic sign detection range is 25 meters in case of good lighting, the average recognition time is 5 seconds. The methods and techniques to increase system performance have been proposed.**

Keywords—computer vision, machine learning, neural network, image recognition, traffic sign recognition, single-board computer

I. Introduction

Owing to the increase in the number of vehicles, ensuring the road safety with the Advanced Driver Assistance System (ADAS) [1] becomes an important field of research [1-3]. The ADAS provides a lot of information about road environment necessary for safe driving [4]. The information includes traffic light signs, traffic signs, road surface marking, location of neighboring cars on the road, etc. An important part of ADAS is the algorithm for recognizing traffic signs of speed limits [1, 3, 5].

The paper is organized as follows. Section II outlines the basic approach to traffic sign recognition. Computer vision algorithms for traffic sign detection are presented in Section III. The machine learning algorithm used for traffic sign recognition is described in Section IV. Section V provides an overview of the hardware for implementing the proposed system. Test results and approaches for increasing performance are presented in Section VI. Finally, conclusions are outlined in Section VII.

II. Structure of the Traffic Sign Recognition System

Traffic Sign Recognition System (TSR) usually contains the following main stages (Fig. 1): preprocessing stage, detection stage, post-processing stage and recognition or classification stage [5-7]. However, sometimes pre- and post-processing stages are supposed to be included in the detection stage. The pre-processing stage implements the necessary preparation on the input image, such as conversion to other color model, scaling, etc. The next stage is a detection of image areas that are likely to contain a traffic sign. Such image areas are usually called the regions of interest (ROI). Then obtained ROIs passed to the post-processing stage, where the operations like the compression of a detected image areas, resizing, normalization, smoothing, etc. are carried out. Such post-processing is determined by the requirements of the next stage. Traffic sign recognition and classification by the predefined categories are performed on the last stage. Ordinarily the computer vision methods are used at the first three stages. Machine learning is used at the last stage.

Since speed limit signs in different countries have differences (background with numbers, signatures on signs, different fonts), it is more efficient to divide the software architecture into two stages:

1) Detection of a speed limit traffic sign by computer vision methods
2) Recognition of digits inside the detected sign by machine learning algorithms

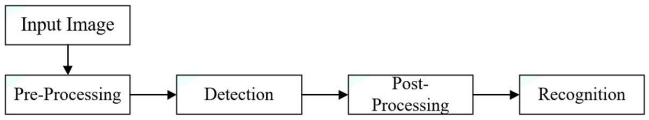

Fig. 1. Flowchart of a typical traffic sing recognition algorithm

The system proposed in this paper works as follows. First, video is captured from the camera, thinned out to select every 5th frame. Further, by computer vision algorithms, the image is pre-processed in order to detect the sign and numbers inside it. Then, after post-processing, the detected numbers are alternately transmitted to the neural network for subsequent recognition. The result is displayed on the image as a selected area of the sign and signature. The software part of the proposed system can be implemented in both compiled and interpreted programming languages. In this work the Python programming language was used.

III. Traffic Sign Detection

Computer vision algorithms for traffic signs detection are directly connected with their unique features. Foremost, all traffic signs have one of the following shapes: triangle, rectangle, circle or octagon [2, 5]. The colors used in the traffic signs are usually ones of the simple colors, such as red, blue, black, white, yellow and less often green [2]. Consequently, the approaches of the detection stage are mostly divided into two groups: color-based methods and shape-based methods [1, 7]. Color-based methods utilize the fact, that traffic signs are designed using a very limited number of colors, which are usually contrasted to the surroundings. Image areas containing these colors are extracted from the input image and then used as a base for detection [5]. In many cases switching from RGB (Red-Green-Blue) to other color model such as HSV (Hue-

Saturation-Value), GRAY (Grayscale model where picture is presented as a single channel model where each pixel is an amount of light) or HIS (Hue-Saturation-Intensity) is performed to exclude the influence of lightness changing. Another frequently used color models are YUV (Luma-Blue Projection-Red Projection) and XYZ (Etalon Color Space Model). Shape-based methods ignore the color component but use such advantage of the traffic signs as well-defined shape [1, 2]. Canny edge detection algorithm is one of the commonly used shape-based methods [8].

External factors such as lighting, weather conditions, sign surface quality, etc. can significantly affect color-based methods [1, 2]. The shape-based methods are less dependent by the lightness, weather or surface quality since the sign shape stays the same. Although, the effectiveness of such methods are affected by occlusion, rotation or location at the background of a similar color [1]. Shape-based detection in urban environment is also not an easy task due to the visual clutter [2]. Which detection method will be used in the development of software is decided by the developer himself, depending on the task to be solved.

There are a number of computer vision frameworks and libraries. Among them the most commonly used is OpenCV, which is suitable for porting directly to embedded systems on microcontrollers, single board computers, etc., thereby facilitating the development of computer vision systems.

In this paper, a shape-based approach was applied to detect a speed limit traffic sign. The following OpenCV algorithms were used at the stage of preprocessing the input image:

- Blurring the original image in RGB color space (Fig. 2) in order to smooth out and minimize noise, and to avoid blobs when searching for outlines: function *cv2.GaussianBlur(image, core (x, x), borderType)*.

Fig. 2. Gaussian blur

- Finding edges and contours (calculating the Laplacian) of an image: function *cv2.Laplacian(img, depth, ksize, scale, delta, borderType)*.
- Conversion from RGB color space to grayscale (Fig. 3). It is often required to implement many functions and conversions: function *cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)*.

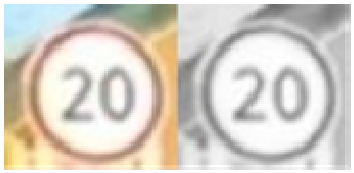

Fig. 3. Converting to grayscale

- In order to simplify the localization of the sign, the circle search function (Fig. 4) is used according to certain parameters: the function *cv2.HoughCircles(img, cv2.HOUGH_GRADIENT, dp, minDistance, param1, param2, minRadius = x, maxRadius = y)*.

Fig. 4. Finding circles

- During image processing, many functions often require converting a color image to a binary one (black and white) (Fig. 5): function *cv2.threshold(gray_img, thresh, val, type)*.

Fig. 5. Image binarization

- To find the contours of numbers the function: *cv2.findContours (img, mode, method)* was used. In addition, for clarity, the contour line drawing function was also used in the work: *cv2.drawContours(img, contours, contourIdx, color = (x, x, x), thickness, lineType)* (Fig. 6).

Fig. 6. Drawing a contour

- Due to the fact that the speed limit traffic sign contains several numbers, and the neural network is designed for the recognition of a single digit, it is necessary to separate them. This mechanism is provided by the function *cv2.rectangle(img, point1, point2, color (x, x, x), lineType)* (Fig. 7).

Fig. 7. Draw a rectangle around the detected digits

IV. Traffic Sign Recognition

A. Machine Learning

Machine learning is a set of techniques and methods for creating artificial intelligence that use algorithms and large amounts of data to simulate the learning process. Learning is carried out according to the general principle: first, training takes place on structured or unstructured data, and then functions are automatically determined that label this data in different categories [9]. Deep learning is a subset of machine learning, which is based on neural network architectures with a large number of inner layers, as, for example, in convolutional neural networks (CNN) used in this work [10].

One of the main features of such neural network is the usage of convolutional layers and pooling layers which are alternated by each other (Fig. 8). Convolutional layers have their own convolution kernel for each data channel, convolve the input and pass its result to the next layer. Pooling layers reduce the dimensions of the data being processed by combining the outputs of neuron clusters at one layer into a single neuron in the next layer. The functioning of a CNN is usually interpreted as a transition from specific features of an image to more abstract details and then to even more abstract details up to the isolation of high-level concepts. As a result, convolutional neural networks are resistant to the shifting of the image being recognized [11, 12].

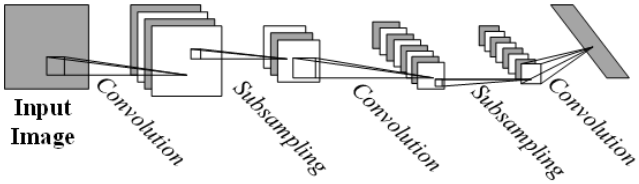

Fig. 8. Convolutional neural network

B. Neural network architecture and training

The neural network model used in this paper consists of 784 neurons at the input, since the image will come in dimensions of 28 by 28 pixels. The following are 2 consecutive convolutional layers that will work with the input image. Then the zero filling will be performed, which is necessary to reduce the image by the filter size, after which the size of the original matrix will be adjusted using the MaxPooling2D downsampling layer. This is followed by a convolutional layer and a downsampling layer. A fully connected layer of 256 neurons is provided in front of the output layer. The purpose of this layer is to provide a multiple feature map extraction system and to reduce the size of the input data. The output layer will have 10 neurons due to the fact that individual numbers will be recognized.

The MNIST dataset from TensorFlow, containing handwritten numbers, was taken to train the model. This dataset contains about 60000 images, divided into 10 categories. Also, to evaluate the model, a test sample of 10,000 images of digit characters segmented from standard license plates was used. In addition, model accuracy testing was carried out on images of various speed limit traffic signs from the GTSRB dataset. This dataset contains images of German traffic signs. The images from this dataset can be used for training and testing the system that recognize traffic signs conforming the Vienna Convention on Road Signs and Signals. This convention has been ratified by 62 countries.

The neural network model was trained using classical categorical cross-entropy. The gradient descent algorithm (RMSprop) is used to redefine the weights after initialization. The number of epochs was set to 7, since this number was sufficient to ensure good accuracy. The batch size was 500 images. The learning outcomes are presented below in Fig. 9-10.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 26, 26, 64)        640
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 24, 24, 128)       73856
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 12, 12, 128)       0
_________________________________________________________________
dropout (Dropout)            (None, 12, 12, 128)       0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 10, 10, 128)       147584
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 5, 5, 128)         0
_________________________________________________________________
flatten (Flatten)            (None, 3200)              0
_________________________________________________________________
dense (Dense)                (None, 256)               819456
_________________________________________________________________
dropout_1 (Dropout)          (None, 256)               0
_________________________________________________________________
dense_1 (Dense)              (None, 10)                2570
=================================================================
Total params: 1,044,106
Trainable params: 1,044,106
Non-trainable params: 0
```

Fig. 9. The architecture of the model

```
Test loss: 0.037336599081754684
Test accuracy: 0.9897000193595886
```

Fig. 10. Results of the trained model test

V. Hardware of The Speed Limit Traffic Sign Recognition System

Single-board computers are suitable for the implementation of an embedded device, the use of which is narrowly limited to the software created for it. Single-board computers (SBC) are specialized microprocessor control systems of a small size. It includes a memory, a processor, a storage device, specialized I/O connector (GPIO), USB ports, Ethernet, HDMI, as well as modules Wi-Fi, Bluetooth, etc. The Raspberry Pi 3 Model B+ (RPi3B+) is one of the most famous representative of the SBC. The board contains a Broadcom BCM2837B0 Cortex-A53 (ARM v8) processor, clocked at 1.4 GHz and a VideoCore IV graphics processor with OpenGL and OpenVG support. The OpenCV library can be installed on this single board computer for pattern recognition using neural networks. The functioning of the board is usually carried out under the control of the Debian distribution – Raspberry Pi OS.

VI. Test Results of The Traffic Sign Recognition System

A. Results

For testing the system proposed in the paper a video file was processed, in which the speed limit sign was recognized, and the resulting value was displayed on the screen. It should be noted that the distance from the camera to the sign is from 10-12 meters to 25 meters depending on the lighting. Observations have shown that in the daytime with direct sunlight on a sign recognition is the most accurate and fastest (for a 20 km/h sign – 5 seconds), with a decreasing of lighting, the efficiency of a sign recognition reduces (for a 50 km/h sign – 6 seconds), at night the sign detection becomes ineffective. The test results of the developed speed limit traffic sign recognition system based on the Raspberry Pi 3B+ single-board computer are shown in Fig. 11-12.

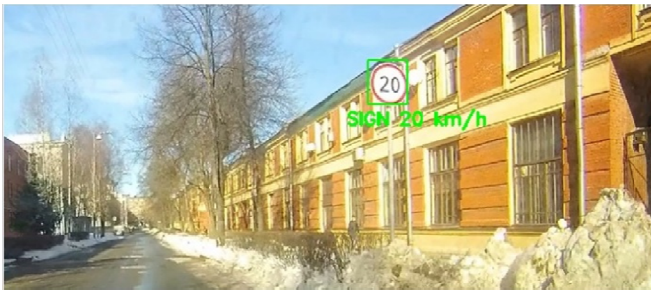

Fig. 11. Sign recognition result 20 km/h

Fig. 12. Sign recognition result 50 km/h

B. Increasing system recognition speed

FPGA can be applied both to speed up image preprocessing and speed up computations performed by a neural network. The OpenCV library is a good option for FPGA development, since it allows to reduce development time while maintaining performance compared to creating a custom algorithm [13]. For example, for Canny's algorithm the image is divided into blocks of 64x64 pixels so that they can be stored in memory. The algorithm is parallelized and run on the FPGA board. Thus, using the ported algorithm and parallelizing the image processing allowed to process a 512x512 image in 0.72 ms on an FPGA board, compared to an initial time of 26.112 ms on a GTX 80 GPU. Table I shows the total processing time for the case using only ARM processor and for acceleration with an FPGA for various image resolutions. The use of FPGA allowed to reduce processing time by 48.2 %, 49.5 % and 56.1 % for resolutions 640x480, 800x600 and 1024x768 respectively [14].

TABLE I. COMPARATIVE CHARACTERISTICS OF ARM PROCESSOR AND FPGA

Resolution	**Tracking Algorithm (OpenCV)**	
	Total processing time / frame rate	
	ARM	***Acceleration using FPGA***
640x480	0.1268s/7.89fps	0.0656s (0.0216s)/15.24fps
800x600	0.2165s/4.62fps	0.1092 (0.0352s)/9.16fps
1024x768	0.3691s/2.71fps	0.1624s (0.0559s)/6.16fps

To increase the performance of the machine learning phase, special hardware accelerators can be used. One of the most common among them is the Intel® Neural Compute Stick 2 (Intel NCS 2) – the device for accelerating development, learning and operation of the neural network on SBCs, including Raspberry Pi, without loss of accuracy. It has a low power consumption of 1 W with a throughput of 1 trillion operations per second (1 TOPS). This device has a USB 3.0 connector that is fully backward compatible with USB 2.0 single-board computer. The OpenVINO libraries are needed to work and communicate with this device.

VII. CONCLUSIONS

In the paper, a prototype of a speed limit traffic sign recognition system based on a Raspberry Pi 3 Model B+ single board computer has been proposed. The shape-based approach was used for speed limit traffic sign detection. Then a convolutional neural network was applied to recognize digits inside the sign. The proposed system in case of good lighting has the recognition time 5 seconds, in case of poor lighting – 6 seconds. The approaches to increasing speed limit traffic sign recognition system performance have been proposed. Using the proposed techniques, the traffic sign recognition speed is expected to be increased by 50-60 %.

REFERENCES

[1] Fan, Y., Zhang, W. Traffic sign detection and classification for Advanced Driver Assistant Systems. In the 12th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD), 2015, pp. 1335-1339.

[2] Oruklu, E., Pesty, D., Neveux, J., Em Guebey, J. Real-time traffic sign detection and recognition for in-car driver assistance systems. In the IEEE 55th International Midwest Symposium on Circuits and Systems (MWSCAS), 2012, pp. 976-979.

[3] Mogelmose, A., Liu, D., Trivedi M. Detection of U.S. traffic signs. In IEEE Transactions on Intelligent Transportation Systems, 2015, vol. 16, issue 6, pp. 3116 – 3125.

[4] Pritt, C. Road sign detection on a Smartphone for Traffic Safety. In IEEE Applied Imagery Pattern Recognition Workshop (AIPR), 2014, pp. 1-6.

[5] Sheikh, A. A., Kole, A., Maity, T. Traffic sign detection and classification using colour feature and neural network. In International Conference on Intelligent Control Power and Instrumentation (ICICPI), 2016, pp. 307-311.

[6] Roxas, E. A., Aclio, J. N., Vicerra, R. R. P., Dadios, E. P., Bandala, A. Vision-based traffic sign compliance evaluation using convolutional neural network. In IEEE International Conference on Applied System Invention (ICASI), 2018, pp. 120-123.

[7] Zhao, T., Liu, Q., Zhang, Y. A traffic sign detection method based on saliency detection. In the 1st International Conference on Industrial Artificial Intelligence (IAI), 2019, pp. 1-6.

[8] Arunmozhi, A., Gotadki, S., Park, J., Gosavi, U. Stop sign and stop line detection and distance calculation for autonomous vehicle control. In IEEE International Conference on Electro/Information Technology (EIT), 2018, pp. 356-361.

[9] Gu J., Wang Z., Kuen J., Ma L., Shahroudy A., Shuai B., Liu T., Wang X., Wang L., Wang G., Cai J., Chen T. Recent Advances in Convolutional Neural Networks. Pattern Recognition, 2018, N 77, pp. 354-377.

[10] Indolia S., Goswami A. K., Mishra S. P., Asopa P. Conceptual Understanding of Convolutional Neural Network - A Deep Learning Approach. Procedia Computer Science, 2018, N 132, pp. 679 – 688.

[11] Xin, R., Zhang, J., Shao, Y. Complex networks classification with convolutional neural network. In the Tsinghua Science and Technology, 2020, vol. 25, issue 4, pp. 447-457.

[12] Rawat, W., Wang, Z. Deep convolutional neural networks for image classification: a comprehensive review. In the Neural Computation, 2017, vol. 29, pp. 2352-2449.

[13] Cortes A., Velez I., Irizar A. High level synthesis using Vivado HLS for Zynq SoC: Image processing case studies. 2016 Conference on Design of Circuits and Integrated Systems (DCIS), 2016, p. 1.

[14] Zhang S. Real Time Image Processing on FPGAs, Liverpool, University of Liverpool, 2018, 206 p.