Neural networks modification for solving the traffic signs detection problem

Devyatkin A.V., Filatov D. M.
Department of ACS, SPbSETU "LETI", St. Petersburg, Russia
E-mail: avdevyatkin@etu.ru, dmfilatov@etu.ru

Received xxxxxx
Accepted for publication xxxxxx
Published xxxxxx

Abstract. The paper deals with the implementation of the traffic signs detection model on the deep neural network basis. The complexity of the problem lies in the detected objects size. The paper considers the use of modern approaches to the deep neural networks Mobile Net v2 and Tiny YOLO v3 implementation to solve the detection problem in real time. Also, a modification of the considered networks is proposed, which allows increasing the Average Precision index by more than 20%. For the networks training GPGPU NVIDIA 1080 and a signs images set called Russian traffic sign images dataset - RTSD which includes more than 15,000 frames for training and more than 3,000 frames for carrying out testing were used.

Keywords: deep neural network, detection, YOLO, MobileNet

1. Introduction

Modern working methods with images and developments in the field of algorithms based on training allow determining various objects with a sufficiently high accuracy and highlighting their location in the frame. In the initial stages, classical methods were used, such as the Haar classifier, the template method, the Viola-Jones method. Today, due to the development of special hardware platforms, modern neural network architectures allow both increasing the accuracy and detection methods speed. Such architectures are SSD [1], YOLO [2], Fast R-CNN [3], Mask R-CNN [4], FPN [5]. Each of the approaches has its own specifics, but the general trend in the recognition problem is testing on generalized data sets specially prepared for evaluation and comparison of detection algorithms. The most common datasets are COCO [6], Pascal VOC [7].

In the field of computer vision for the unmanned vehicles creation, the researchers highlight one of the important tasks - traffic signs detection, which allows the control system when using unmanned mode or driver assistance mode to identify signs as objects with information and take this information into account when planning or controlling the traffic trajectory. Despite the fact that in the future, with full road traffic automation, the need for the road signs use may completely disappear in the coming years, when on public roads vehicles with both autonomous control and under human control will be used, the problem of road signs recognition will not lose its relevance.

To solve the detecting signs problem with the use of methods based on training, special data sets are used, which contain marked data corresponding to the task specifics. In the development of the road signs detector such dataset as German set of signs [8], Belgian set of signs [9], Swedish set of signs [10] and others can be used.

The similarity of general and specialized datasets with road signs is that the annotation is made with the area indication in the image. However, the road signs detecting task is complicated by the fact that the size of the sign on the frame is often relatively small compared to the General dataset, where objects can occupy half a frame. The sign image can occupy several tens of pixels in width per Full HD image, which determines the specifics of this task.

2. Neural networks for detection
Recognition approaches based on deep neural networks can be evaluated by a number of indicators. In this paper two of the most interesting indicators are identified:
- Operation speed,
- Detection accuracy.

Previously, the development trends were aimed at increasing the detection methods accuracy, but today the hardware of the calculations allows getting the work of methods in real time, which is especially important when driving unmanned vehicles. The following two architectures stand out as the solutions: SSD and YOLO. Both architectures are one-pass detection methods, for which the input is the image, and the result is predictions on the object localization with the indication of the class in the image.

This paper uses the Yolo 3 architecture [2] as one of the real-time detectors with the classic Tiny configuration (a smaller version of the original network). Also for comparison and evaluation the basis of Tiny YOLO is replaced by MobileNet2 architecture [11], which consists of special blocks aimed at reducing the number of parameters for network operation, which has a positive impact on learning and memory consumption.

YOLO architecture can be divided into two blocks:
- the basic convolution unit,
- the output prediction layer(-s).

The main convolution unit is the main component for the selection and processing of features in the image. The number of filters in the prediction output layer and the size of the layer are set according to the number of predicted feature classes and the number of cells into which the image is split.

According to this view, in the classic Tiny YOLO architecture, it is possible to replace the main feature allocation unit with the MobileNet2 architecture network. A summary view of the architectures is presented in tables 1 and 2. Layer designations are deciphered as follows:
- Conv-Convolution,
- Concat-Concatenation,
- CBR-Convolution + Batch Normalization + Relu,
- MP-Max Pooling,
- US - Up sampling,
- IRB - Inverted residual block [11]

Table 1. Tiny YOLO network description

| # | Input# | Layer, output size | Number of filters, the size of the convolution kernel |
|---|--------|--------------------|------------------------------------------------------|
| 0 | Input, 416x416 | | |
| 1 | 0 | CBR + MP, 208x208 | 16, 3 |
| 2 | 1 | CBR + MP, 104x104 | 32, 3 |
| 3 | 2 | CBR + MP, 52x52 | 64, 3 |
| 4 | 3 | CBR + MP, 26x26 | 128, 3 |
| 5 | 4 | CBR + MP, 13x13 | 256, 3 |
| 6 | 5 | CBR, 13x13 | 512, 3 |
| 7 | 6 | CBR, 13x13 | 1024, 3 |
| 8 | 7 | CBR, 13x13 | 256, 1 |
| 9 | 8 | CBR, 13x13 | 512, 3 |
| 10 | 9 | Conv, 13x13 | 18, 1 |
| 11 | 8 | CBR + US, 26x26 | 128, 1 |
| 12 | 11, 4 | Concat + CBL, 26x26 | 256, 3 |
| 13 | 12 | Conv, 26x26 | 18, 1 |
Table 2. Mobile Net 2 description

| #  | Input# | Layer, output size | Number of filters, the size of the convolution kernel |
|----|--------|--------------------|-----------------------------------------------------|
| 0  |        | Input, 416x416     |                                                     |
| 1  | 0      | CBR + MP, 208x208   | 32, 3                                               |
| 2  | 1      | IRB, 208x208       | 16, 3                                               |
| 3  | 2      | IRB x2, 104x104    | 24, 3                                               |
| 4  | 3      | IRB x3, 52x52      | 32, 3                                               |
| 5  | 4      | IRB x4, 26x26      | 64, 3                                               |
| 6  | 5      | IRB x3, 26x26      | 96, 3                                               |
| 7  | 6      | IRB x3, 13x13      | 160, 3                                              |
| 8  | 7      | IRB, 13x13         | 320, 3                                              |
| 9  | 8      | CBR, 13x13         | 1280, 1                                             |
| 10 | 9      | Conv, 13x13        | 18, 1                                               |

In this paper, training based on the Russian roads signs set (RTSD) is used [12]. This dataset contains more than 15,000 frames from the road for training and more than 3,000 frames for testing algorithms. For training, the Adam method was used with a learning factor of 0.001. Data expansion was performed using augmentation by the following methods:
- random shift,
- random turn,
- random horizontal reflection,
- normally distributed noise,
- color distortions in HSV space.

For evaluation purposes, the Average Precision (AP) metrics are used with a threshold value of the Jakarta (or Intersection Over Union) equal to 0.5 and the speed of work in frames per second (FPS) of the network on the test data set. The input image size for both architectures is 416x416 pixels. Both network configurations use three reference rectangles per output layer. The Tiny YOLO network uses output layers of size 13 and 26, while MobileNet 2 uses a standard output layer of size 13. The number of output layers is set according to the formula:

\[ F = N \times (4 + 1 + C), \]

where \( F \) is the number of prediction layer filters, \( N \) is the number of reference rectangles per output layer, and \( C \) is the number of predicted classes. In this case: \( C = 1, N = 3, F = 18 \).

The performance indicators of trained networks are presented in table 3.

Table 3. Network accuracy indicators

| Architecture type | AP, % | FPS |
|-------------------|-------|-----|
| Tiny YOLO         | 30,8  | 96,9|
| MobileNet2        | 29,6  | 101,3|

As can be seen from the indicators presented in table 3, the proposed standard network configurations, focused on a common set of medium and large objects, cope with the detection task with low accuracy.

3. Network modification and results

Based on the idea that the minimum cell size of the output grid in the Tiny YOLO network is 16 pixels, it can be assumed that adding outputs from the 8 pixel grid sizes should increase the resolution of the model and increase the accuracy of detecting small objects, as this will allow the model to analyze the input image in more detail.
To improve the performance, the following modifications of the source architectures are proposed to improve detection accuracy (tables 4 and 5).

Table 4. Tiny YOLO modification for small objects

| #  | Input # | Layer, output size       | Number of filters, the convolution kernel size |
|----|---------|--------------------------|----------------------------------------------|
| 0  |         | Input, 416x416           |                                              |
| 1  | 0       | CBR + MP, 208x208        | 16, 3                                        |
| 2  | 1       | CBR + MP, 104x104        | 32, 3                                        |
| 3  | 2       | CBR + MP, 52x52          | 64, 3                                        |
| 4  | 3       | CBR + MP, 26x26          | 128, 3                                       |
| 5  | 4       | CBR + MP, 13x13          | 256, 3                                       |
| 6  | 5       | CBR, 13x13               | 512, 3                                       |
| 7  | 6       | CBR, 13x13               | 1024, 3                                      |
| 8  | 7       | CBR, 13x13               | 256, 1                                       |
| 9  | 8       | CBR, 13x13               | 512, 3                                       |
| 10 | 9       | Conv, 13x13              | 18, 1                                        |
| 11 | 8       | CBR + US, 26x26          | 128, 1                                       |
| 12 | 11, 4   | Concat + CBL, 26x26      | 256, 3                                       |
| 13 | 12      | Conv, 26x26              | 18, 1                                        |
| 14 | 11      | CBR + US, 52x52          | 128, 1                                       |
| 15 | 14, 3   | Concat + CBL, 52x52      | 128, 3                                       |
| 16 | 15      | Conv, 52x52              | 18, 1                                        |

Table 5. MobileNet 2 modification for small objects

| #  | Input # | Layer, output size       | Number of filters, the convolution kernel size |
|----|---------|--------------------------|----------------------------------------------|
| 0  |         | Input, 416x416           |                                              |
| 1  | 0       | CBR + MP, 208x208        | 32, 3                                        |
| 2  | 1       | IRB, 208x208             | 16, 3                                        |
| 3  | 2       | IRB x2, 104x104          | 24, 3                                        |
| 4  | 3       | IRB x3, 52x52            | 32, 3                                        |
| 5  | 4       | IRB x4, 26x26            | 64, 3                                        |
| 6  | 5       | IRB x3, 26x26            | 96, 3                                        |
| 7  | 6       | CBR, 26x26               | 1280, 1                                      |
| 8  | 7       | Conv, 26x26              | 18, 1                                        |
| 9  | 6       | US, 52x52                |                                              |
| 10 | 9, 4    | Concat + IRB x2, 52x52   | 32, 3                                        |
| 11 | 10      | CBR, 52x52               | 640, 1                                       |
| 12 | 11      | Conv, 52x52              | 18, 1                                        |
As can be seen, in the presented modifications Tiny YOLO has output layers with the size of 13, 26 and 52 cells, and MobileNet2 - only 26 and 52 cells, omitting the possibility of analyzing the image using a grid with 13 cells. This approach assumes that the modified MobileNet2 architecture will weakly detect large objects that are not present in the data set as such.

The learning outcomes and assessments in the test sample are presented in Table 6.

Table 6. Accuracy indicators of the modified networks

| Architecture types | AP, %  | FPS   |
|--------------------|--------|-------|
| Tiny YOLO          | 48.7   | 48.2  |
| MobileNet2         | 66.0   | 45.9  |

As can be seen from the indicators, the modification of the Tiny YOLO network allowed to greatly increase the accuracy of the network operation with a corresponding decrease in the operation speed. The presence of an output layer with a size of 13x13, focused on the analysis of sufficiently large objects, presumably reduces the model efficiency, which is confirmed by greater accuracy in the modification MobileNet2, which has no large objects analysis.

Decrease in the speed of both modifications does not exceed the limits of real time operation and is relatively higher than the speed of most modern cameras, for this reason they can be considered acceptable.

4. Conclusion

This paper presents an approach to modify the deep neural network architecture in order to increase the detection accuracy when working with small objects. The presented results confirm the hypothesis that increasing the resolution of the output layer grid (the output layer size) increases the ability of the network to analyze small objects, which is particularly important in working with remote traffic signs on the road. The considered approach allows to apply General purpose neural networks and to increase detection accuracy at work with certain ranges of detectable objects.

Further work involves the neural networks study in terms of the number of filters parameters and sizes in order to compensate for the loss in speed without accuracy loss in detecting traffic signs.

References

[1] Liu, Wei et al. “SSD: Single Shot MultiBox Detector.” Lecture Notes in Computer Science (2016): 21–37. Crossref. Web.

[2] J. Redmon and A. Farhadi. Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767, 2018.

[3] Ross Girshick. Fast R-CNN. arXiv preprint arXiv:1504.08083, 2015.

[4] Kaiming He and Georgia Gkioxari and Piotr Dollár and Ross Girshick. Mask R-CNN. arXiv preprint arXiv:1703.06870, 2017.

[5] Tsung-Yi Lin and Piotr Dollár and Ross Girshick and Kaiming He and Bharath Hariharan and Serge Belongie. Feature Pyramid Networks for Object Detection. arXiv preprint arXiv:1612.03144, 2016.

[6] Lin, Tsung-Yi et al. “Microsoft COCO: Common Objects in Context.” Lecture Notes in Computer Science (2014): 740–755. Crossref. Web.

[7] Mark Everingham, Luc Gool, Christopher K. Williams, John Winn, and Andrew Zisserman. 2010. The PASCAL Visual Object Classes (VOC) Challenge. Int. J. Comput. Vision 88, 2 (June 2010), 303-338. DOI=http://dx.doi.org/10.1007/s11263-009-0275-4

[8] J. Stallkamp, M. Schlipsing, J. Salmen, C. Igel, Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition, Neural Networks, Available online 20 February 2012, ISSN 0893-6080, 10.1016/j.neunet.2012.02.016. (http://www.sciencedirect.com/science/article/pii/S0893608012000457)

[9] Radu Timofte*, Markus Mathias*, Rodrigo Benenson, and Luc Van Gool, Traffic Sign Recognition - How far are we from the solution?., International Joint Conference on Neural Networks (IJCNN 2013), August 2013, Dallas, USA.

[10] Fredrik Larsson and Michael Felsberg , Using Fourier Descriptors and Spatial Models for Traffic Sign Recognition , In Proceedings of the 17th Scandinavian Conference on Image Analysis, SCIA 2011, LNCS 6688, pp. 238-249. bib , doi:10.1007/978-3-642-21227-7_23
[11] Mark Sandler and Andrew Howard and Menglong Zhu and Andrey Zhmoginov and Liang-Chieh Chen. MobileNetV2: Inverted Residuals and Linear Bottlenecks. arXiv: 1801.04381. 2018.
[12] Shakhuro, V.I. & Konushin, Anton. (2016). Russian traffic sign images dataset. Computer Optics. 40. 294-300. 10.18287/2412-6179-2016-40-2-294-300.