Development of Traffic Signs Recognition WebService based on Convolutional Neural Networks

K A Pronchuk[1] and P Y Yakimov[1]

[1]Samara National Research University, Moskovskoye Shosse 34A, Samara, Russia, 443086

Abstract. Image classification is one of the most important applications of neural networks. In this paper,we propose the classification algorithm for traffic signs recognition based on convolutional neural networks.The designed CNN is implemented using the TensorFlow framework, and the inference is performed using CUDA. To utilize the Connected Cars concept, we also developeda webservice to remotely process images, obtained by a camera installed into a vehicle. The experimental results show that the proposed algorithm, implemented using the CUBA library for developingclient-server apps, shows high efficiency and is applicable for a connected vehicle.

1. Introduction

The Advanced driver assistance systems (ADAS) are high-tech vehicle systems that are designed to improve road safety, help drivers better understand the road and its potential dangers. The recognition of road signs is one of the important subsystems of ADAS. Implementing a real time traffic sign recognition system is usually divided into two steps: localization of a road sign and its classification. The detection is often performed using simple computational algorithms, such as the color thresholding with shape detection [1], [2], [3]. After that, the classification takes place using more complex, and at the same time more accurate algorithms.

Recently, with the development of high speed networks like 5G, the concept of Connected Cars [4] is becoming more popular and, more important, more possible.The term ‘Connected Car’ refers to applications, services, and technologies that connect a vehicle to its surroundings. A connected vehicle is basically the presence of devices in a vehicle that connect to other devices within the same vehicle and/or devices, networks, applications, and services outside the vehicle. The applications of the concept include everything from traffic safety and efficiency, infotainment, parking assistance, roadside assistance, remote diagnostics, and telematics to autonomous self-driving vehicles and global positioning systems (GPS). Typically, vehicles that include interactive advanced driver-assistance systems and cooperative intelligent transport systems (C-ITS) can be regarded as connected.

Usually, on-board computers of connected vehicles have rather low performance, which is quite enough to implement various network connection functions, including sending the data from its sensors and receiving some data from other vehicles.But using it for such tasks as traffic sign recognition is impossible. However, nowadays it becomes possible to use remote computing power to solve this problem: the service can send the image to a remote server for processing and further obtaining a response about the presence of road signs in it. Thus, in the traffic sign recognition task, the web service only transfersan image from the camera installed into a vehicle to remote server, gets

the response from it and displays it on an interactive map, which then can used by other cars in the whole system.

2. Recognition algorithm

2.1. HAAR cascade classifier for sign detection

The Haar feature-based Cascade Classifier initially proposed by Viola and Jones [5], [6] classifies objects using a series of edges, lines, and center-surround features that scan throughout the image to construct ROI features. The name cascade means that the resultant classifier consists of several simpler classifiers that are applied to the image one at a time in the order of their classification effectiveness.To speed up the detection process, when earlier classifiers fail to produce a match, the remaining classifiers are no longer applied.We implemented Haar cascade detection for all 43 classes of signs. The number of classes was chosen because we used a dataset containing 43 types of signs.

2.2. Convolutional Neural Neworks

Recently, neural networks have been actively used in classification tasks. The neural network is a mathematical model built on the principle of the organization and functioning of biological neural networks. Neurons are organized into layers: input, hidden and output. The input layer does not consist of complete neurons, but rather consists of values that are inputs to the next layer. The next layer is hidden. In one neural network, there may be several hidden layers. The last level is the output layer, where each class corresponds to one node [7].

The key part to understand, which distinguishes CNN from traditional neural networks, is the convolution operation. Having an image at the input, CNN scans it many times to look for certain features. This convolution can be set with 2 main parameters: stride and padding type.

2.3. Proposed implementation

In this paper, we chose a classic LeNet model which contains only two convolution layers. To train the model we use TensorFlow – the deep training library. Training and testing were performed on the data set of The German Traffic Sign Recognition Benchmark [6]. In this version of the system, it recognizes the 43 types of traffic signs.

Table 1 shows the architecture of the developed convolutional neural network. The network is a sequence of interconnected layers, starting with a convolutional layer and ending with a layer of softmax. The parameter of the convolutional layer - step - determines the step of the rolling convolution window. In case the step is greater than 1, the convolutional layer will combine with polling. The softmax layer performs the normalization of the results of the previous layer in such a way that its probabilities will form the probabilities of the object's relation to one of 43 classes.

Table 1. LeNet-5 architecture.

Layer	Description
Input	32x32x1 Greyscale image
Convolution 5x5	1x1 stride, valid padding, outputs 28x28x6 + ReLU
Avg pooling	2x2 stride, outputs 14x14x6
Convolution 5x5	1x1 stride, valid padding, outputs 10x10x16 + ReLU
Avg pooling	2x2 stride, outputs 5x5x16
Fully connected	inputs 400, outputs 120 + ReLU
Fully connected	inputs 120, outputs 84 + ReLU
Fully connected	inputs 84, outputs 43
Softmax	

The model can be divided into 2 blocks:the convolution block and the fully connected block. TensorFlow includes network mapping tools that allow you to visualize the model at different levels of abstraction, down to low-level mathematical operations.

To generate so-called augmented data, we randomly chose images to copy. To provide additional information to the model, we randomly rotated this copy and changed its brightness. To implement all these operations,the OpenCV library was used. We performed these operations until each label had 3200 samples. This increased the training set to 139.148 samples. As an illustration, here's a drawing of a sample traffic sign with generated images (rotated and different brightness).

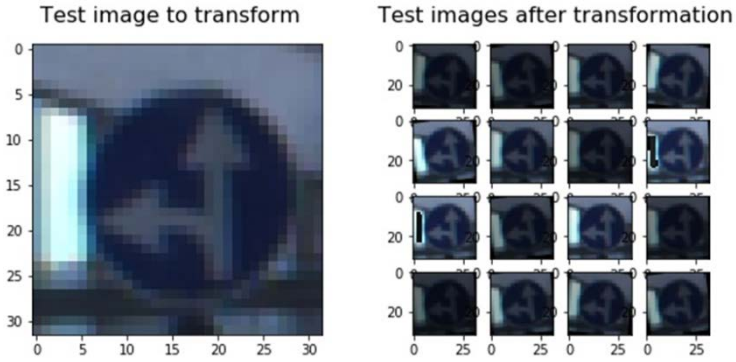

Figure 1. Image transformation examples.

Further, there are the details of intermediate steps and the corresponding validation accuracies after 100 epochs of training.

- Initial LeNet model, choosing input images color representation - 91 %
- Input images normalization - ~91 %
- Training set augmantation - 93 %
- Learn rate optimization, from this stage I tested for 100 epochs - 95 %
- Finding optimum image transformations during training set augmentation - 96 %
- Trying different pool methods, trying dropout, choosing L2 loss, tuning learn rate again - 96.8

Final model results were as follows:

- Training set accuracy of 99.5 %
- Validation set accuracy of 96.8 %
- Test set accuracy of 94.6 %

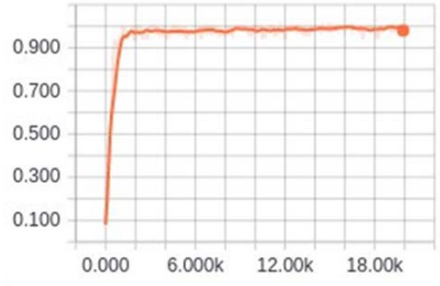

Figure 2. LeNet-5 accuracy.

The convolutional networks also have drawbacks, one of which is the max-pooling layer and the invariance of detection only to the position in the image. The network can falsely recognize the same type of image with a specific rotation, so training is accompanied by an increasing test data and the addition of a max-pooling layer. The layer is used to reduce the dimensions of the outputs of convolutional layers, ignoring small differences in the spatial structure of images, and some information is lost.

In the past decade, the CNNs have really become a revolutionary tool for computer vision developers. However, some new types of neural networks are still introduced. Let's take a closer lookat the architecture of the capsular neural networks described by Hinton in [8]. Figure 3 shows an example architecture of a capsular neural network.

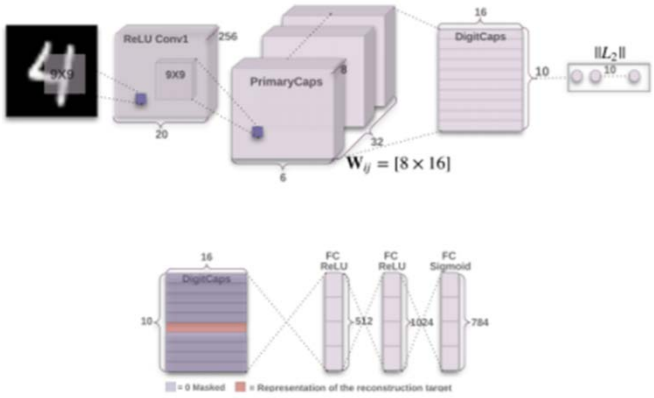

Figure 3. CapsNet architecture.

First, there is a standard separation of features that are invariant to translation in an image, using a convolutional layer. Then another 32 features layers are isolated, each of which is connected to the first convolutional layer. This is necessary for dynamic routing. Next comes the "digitCaps" layer,thelayer that gave the name for the whole network. Each capsule has a strict meaning: the amplitude of the vector in each of them corresponds to the probability of the presence in the image of one of the desired classes. At the bottom of the image, an important part of the architecture is represented: a decoder, on the input of which a layer of DigitCaps is supplied, with all the capsules except the one with the largest amplitude being reset. Decoders are essentially an inverted ANN (artificial neural network), at the input of which a small vector was applied, and at the output - the original image. This decoder provides the second most important part of this architecture: the contents of one capsule should fully describe the subset of a particular class being fed to the input. Otherwise, the decoder can not restore the original image in any way. There is another useful property of the decoder - this is the method of regularization. Close codes in the capsules will be close to Euclid images.

In this paper, we also describe a capsular neural network,trained and then used for signs recognition. The network was trained on a 39,209 images database and contains 43 traffic sign classes. Figure 4 shows an increase in the accuracy of classification over time.

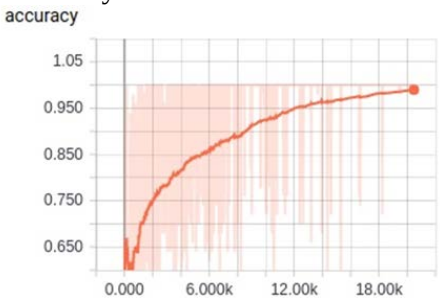

Figure 4. CapsNet accuracy.

3. Web service development

To use the trained neural network, we developeda web service based on the CUBA platform. CUBA is the platform for rapid development business applications in Java, which allows to quickly create a UI and logic for working with data. The Vaadin framework is used in the default web client to display the generic user interface. Vaadin offers a comprehensive set of extensible components and supports SCSS for UI customization. At the same time its server-based implementation model improves security and allows to use Java for both client and server-side code. Spring container provides core infrastructure for the middle tier and the application's client blocks. The framework is also used to establish remote interaction between blocks and for web services implementation.

Shown in figure 5, CUBA Studio is a specialized tool for rapid application development, that streamlines building applications upon CUBA Platform. This is a separate application, which the

developer uses in parallel with the regular Java IDE. The studio provides a graphical interface to the platform mechanisms that allows you to create a data model using the mouse, generate DDL scripts for the database, draw screens in the WYSIWYG editor, and make the middle layer services.

Applications based on CUBA have a standard three-layer architecture. The connecting element of the system is metadata - information about the data model of the application. Relates to metadata, visual components know what data they are working with. In the same way, metadata helps the visual components to work with the database through the ORM, specifying the graphs of objects that need to be downloaded or updated. The same principle applies to the security subsystem, the generation of reports and other parts of the platform.

Applications created with the CUBA platform can be deployed (figure 6) in various configurations, from starting all the components of the application on a single server and up to configurations that provide high fault tolerance, with separated web, middle and database layers. Out of the box supports PostgreSQL, Oracle Database, Microsoft SQL Server, MySQL and HSQL (the latter is usually used for prototyping), between which you can switch as the project grows. It is also important to note that CUBA applications can be deployed on any Java EE Web Profile server, including Jetty, Tomcat, Glassfish, Websphere, etc.

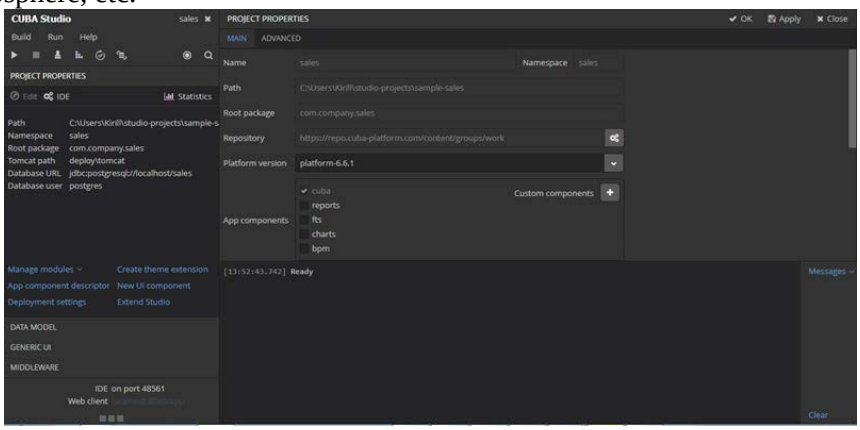

Figure 5. Project main page.

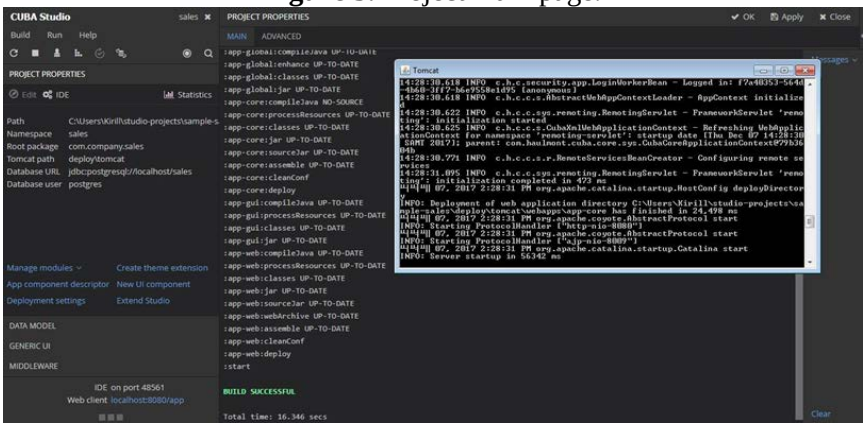

Figure 6. Project deployment.

4. Experiments

The German database of annotated images containing road signs was used [3] to test the accuracy of the detection algorithm. It contains more than 50,000 images with traffic signs registered in various conditions. To assess the quality of detection, the number of images with correctly localized and classified traffic signs was calculated. The experiments showed 96.8% of correctly localized and classified prohibitory and warning road signs.

To test the performance of the developed web service we used images containing a 30 km / h speed limit sign. Various stages of the road sign classification process, such as: loading the trained Tensorflow model, loading a cascade classifier and detecting the presence of a sign on the image,

classifying the sign (if present on the image), the total classification time were evaluated during the experiment.

Table 2. LeNet-5 architecture.

	TF model loading (*ms*)	Sign detection(*ms*)	Sign classification(*ms*)	Total time(*ms*)
1	101	142	12	300
2	113	134	1	315
3	109	157	10	330
4	117	149	12	319
5	100	124	10	423
6	-	-	-	-

Figure 7 shows an example of a successful recognized traffic sign. The information about the sign type can now be transferred to all connected vehicles specifyingits geographical coordinates. Figures 7 and 8 shows the user interface for the developed web service.

Figure 7. Example of recognized image.

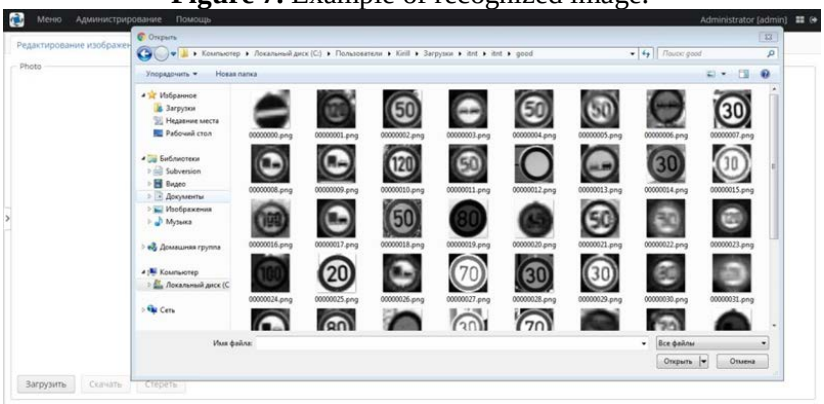

Figure 8. Web service UI.

5. Conclusion

In this project, we developed a method for the classification of traffic signs using convolutional neural networks and integrated it into web service based on CUBA Platform.

The experiments have shown high classification efficiency of 96.8%. Moreover, capsular neural networks showed better result compared with convolutional neural networks. This became possible thanks to the more complex CapsNet device, but the performance also decreased. In final version of application, we usedthe LeNet-5.

In addition to solving the problem of classification of traffic signs, this paper also proposes a method for their localization using the cascade classifier based on the Haar features. This method has successfully proven itself to high productivity and accuracy, which was also confirmed in this work.

The developed web service is intended for use in the Connected Car concept. This service makes it possible to create and fill interactive maps, inform other vehicles about road events, and later can be used for unmanned vehicles.

In future work, we plan to refine the existing solution by adding REST API and the ability to classify characters on video recordings.

6. References

[1] Shustanov A and Yakimov P 2017 CNN Design for Real-Time Traffic Sign Recognition *Procedia Engineering* **201** 718-725
[2] Ruta A, Porikli F, Watanabe S and Li Y 2011 In-vehicle camera traffic sign detection and recognition *Mach Vis Appl.* **22** 359-375 DOI: 10.1007/s00138-009-0231-x
[3] Belaroussi R, Foucher P, Tarel J-P, Soheilian B, Charbonnier P and Paparoditis N 2010 Road Sign Detection in Images: A Case Study *Pattern Recognition 20th International Conference ICPR* 484-488
[4] Koesdwiady A, Soua R and Karray F 2016 Improving Traffic Flow Prediction with Weather Information in Connected Cars: A Deep Learning Approach *IEEE Trans Veh Technol*. 9508-9517
[5] Yakimov P Y 2015 Tracking traffic signs in video sequences based on a vehicle velocity *Computer Optics* **39(5)** 795-800
[6] Houben S, Stallkamp J, Salmen J, Schlipsing M and Igel C 2013 Detection of traffic signs in real-world images: The German traffic sign detection benchmark *Proceedings of the International Joint Conference on Neural Networks* 715-722
[7] Sermanet P and LeCun Y 2011 Traffic sign recognition with multi-scale Convolutional Networks *Neural Networks The International Joint Conference (IJCNN)*
[8] Hinton G E, Sabour S, Frosst N 2017 Dynamic Routing Between Capsules *Proceedings of the 30th Conference on Neural Information Processing Systems (NIPS)*

Acknowledgments
This work was supported by the Russian Foundation for Basic Research - Project # 16-37-60106 mol_a_dk.