Static gestures classification using Convolutional Neural Networks on the example of the Russian Sign Language

Oleg Potkin and Andrey Philippovich

Moscow Polytechnic University, Moscow, 107023, Russian Federation,
WWW home page: http://mospolytech.ru

Abstract. The article describes the dataset developed for the classifier training, the algorithm for data preprocessing, described an architecture of a convolutional neural network for the classification of static gestures of the Russian Sign Language (the sign language of the deaf community in Russia) and represented an experimental data.

Keywords: deep neural networks, convolutional neural networks, classification, gestures, Russian Sign Language

1 Introduction

Human-computer interaction interfaces are diverse in their scope and implementation: systems with console input-output, controllers with gesture control, brain-computer interfaces [1] etc. Systems with the data input based on the custom gestures recognition have gained wide popularity since 2010 after the release of the contactless game controller "Kinect" from Microsoft. These devices increase their market share to this day and become a part of everyday life of different user categories. So, for example, the Volkswagen car manufacturer introduced the multimedia system Golf R Touch Gesture Control for the car multimedia system management using gestures [2], researchers from Cybernet Systems Corporation have developed personal computer software that allows to use hands gestures instead of the usual input devices [3] and Russian scientists have demonstrated a technology of protection against spam bots based on the CAPTCHA mechanism [4].

To convert gestural commands into the control signal, a gesture classification mechanism is needed, which in turn can be obtained from various devices: special gloves defining the coordinates of the joints [5], as well as 2D and 3D video cameras. The approach using gloves has a significant drawback the user needs to wear a special device connected to the computer. However, the approach based on the concept of computer vision using video cameras is considered more natural and less expensive.

The present study demonstrates the system of the static gestures classification for the Russian Sign Language based on the computer vision approach using convolutional neural networks. The topic is relevant and represents a starting point for researchers in the field of gesture recognition.

2 Dataset description for classifier training

For training, cross-validation and testing of the classifier, a set of data was developed[1], containing 1042 images of hands in a certain gesture configuration the class. In total, 10 classes are represented in the data set, each of which corresponds to a certain static gesture of the Russian Sign Language.

Fig. 1 shows sample images from each of the data set classes.

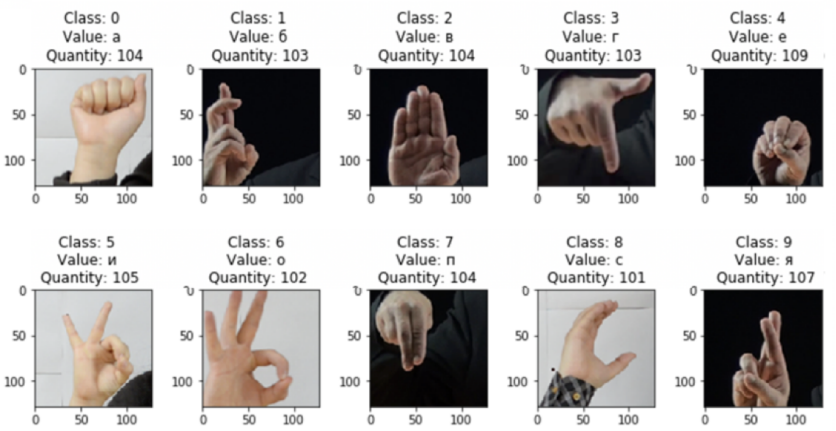

Fig. 1. Samples of images from the dataset. Parameters: the number of the class, the correspondence of the gesture to the letter of the Russian alphabet, the number of images in the class

Images were made with different light conditions and on a small range of object distances from the camera (0.4 0.7 meters).

Table 1 represents the image parameters from the dataset.

Table 1. Parameters of images from dataset

Parameter	Value
Format	.PNG
Width (px)	128
Height (px)	128
Color space	RGB
Color Depth	3

[1] Dataset and the source code of the project are available on GitHub: https://github.com/olpotkin/DNN-Gesture-Classifier

Before sending the data to the classifier, it is necessary to perform image transformations that will help to get rid of minor characteristics for the classifier, which in turn will increase the performance of the classifier. This process is called preprocessing. Image preprocessing reduces the number of unimportant details (e.g. color information from RGB space). Practices and an efficiency of image preprocessing techniques is demonstrated in the publications [6] [7] [8].

In the developed system, the RGB color space was transformed into YCbCr for skin segmentation [9] and binarization was applied to the threshold values (Fig. 2).

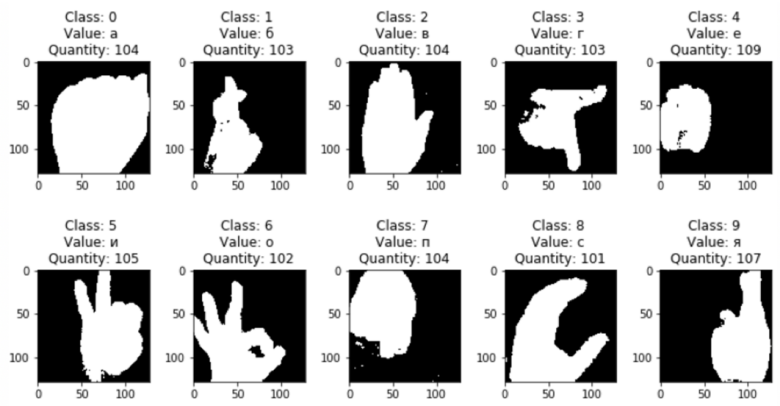

Fig. 2. Samples of images after preprocessing. Parameters: the number of the class, the correspondence of the gesture to the letter of the Russian alphabet, the number of images in the class

The final stage of processing is normalization. The pixel values are in the range from 1 to 255, but for correct operation of the neural network, it is necessary to input values from 0 to 1. To do this, performed a conversion using the formula 1:

$$N = \frac{P}{P_{max}} \tag{1}$$

where N the pixel value after normalization, P the value of the normalized pixel, P_{max} the maximum value of the range.

In order to minimize the overfitting possibility of the classifier [10], the dataset is divided into 3 parts:

- Training set contains 80% of the dataset (666 images).
- Test set, contains 20% of the data set (209 images). Using for evaluation of the model. These samples never used during training and cross-validation processes.
- Cross-validation set, contains 20% of the training sample (167 images).

3 Neural network architecture and the classification results

In this section described CNN architecture on the abstract level. As a benchmark model for an experiment LeNet-5 CNN architecture was taken. The main disadvantage of LeNet-5 is overfitting in some cases and no built-in mechanism to avoid this [11]. So, benchmark architecture was improved by adding dropout layers [13]. Improved LeNet-5 architecture is shown on Fig. 3.

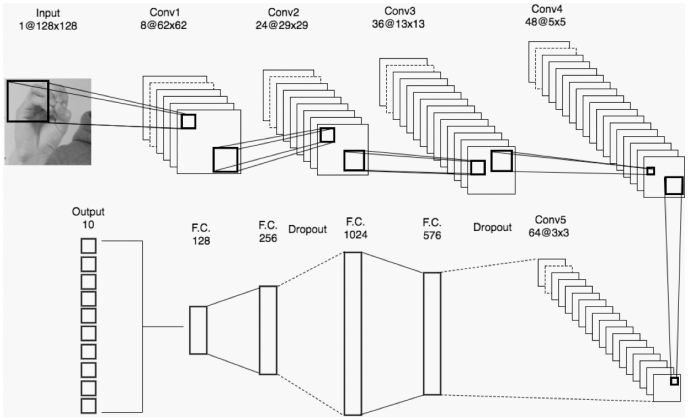

Fig. 3. The architecture of a convolutional neural network for the gestures classification of the Russian Sign Language

The neural network consists of the following layers:

- Input layer (Input). The size of the layer corresponds to the image size after the preprocessing: $1 \times 128 \times 128$.
- Convolution layer (Conv1). Output size: $8 \times 62 \times 62$. Convolution window size: 5×5. Activation function ReLU (Rectified Linear Unit). ReLU can be described by the equation $f(x) = max(0, x)$ and implements a simple threshold transition at zero. Compared to the sigmoid activation function, ReLU increases learning speed and classifier performance [12].
- Convolution layer (Conv2). Output size: $24 \times 29 \times 29$. Convolution window size: 5×5. Activation function ReLU.
- Convolution layer (Conv3). Output size: $36 \times 13 \times 13$. Convolution window size: 5×5. Activation function ReLU.
- Convolution layer (Conv4). Output size: $48 \times 5 \times 5$. Convolution window size: 5×5. Activation function ReLU.
- Convolution layer (Conv5). Output size: $64 \times 3 \times 3$. Convolution window size: 3×3. Activation function ReLU.
- Dropout layer 1, $p = 0.25$. The layer is constructed in such a way that each neuron can fall out of this layer with probability p, therefore, other neurons

remain in the layer with probability $q = 1p$. The dropped neurons are not included in the classifier training, that is, at each new epoch, the neural network is partially changed. This approach effectively solves the overfitting problem of the classifier [13].
- Fully connected layer 1. Size: 576.
- Fully connected layer 2. Size: 1024.
- Dropout layer 2, $p = 0.25$.
- Fully connected layer 3. Size: 256.
- Fully connected layer 4. Size: 128.
- Output layer. Size: 10. Activation function softmax.

The neural network is developed using the Keras framework based on TensorFlow with the following hyper-parameter values after the optimization procedures: the learning rate is 0.0001, the batch size is 128, the number of epochs is 24, the metrics accuracy, the optimizer type Adam.

Training and test procedures were performed on the benchmark classifier (LeNet-5) with results shown in Table 2.

Table 2. Benchmark classifier results

Set	Loss	Accuracy
Training	0.0681	0.9985
Test	0.5134	0.8708

Benchmark model is overfitting because during the training process it shows a way better result compared with test process.

Results of improved architecture with optimization of hyper-parameters shown in Table 3.

Table 3. Classifier optimization results

Set	Loss	Accuracy
Training	0.1411	0.9369
Test	0.2385	0.9138

4 Conclusion

As a result of the research, the dataset consisting of more than a thousand elements belonging to ten different classes of the Russian Sign Language was

developed and published. Algorithms and procedures for preliminary data processing in Python 3.5 were developed. The classifier based on the convolutional neural network using the Keras and TensorFlow frameworks was designed and developed.

The classifier demonstrates an accuracy of classification in 91.38% on the test dataset that is better than result of benchmark classifier 87.08%. The represented results of an accuracy are sufficient to be a strong basis for further research in this direction.

References

1. Potkin, O., Ivanov, V.: Neurointerface Neurosky: interaction with the hardware computing platform Arduino. 64th Open Student Scientific and Technical Conference, Moscow: Moscow State University of Mechanical Engineering, pp. 151-153 (2014)
2. Volkswagen: Gesture Control: How to make a lot happen with a small gesture. [http://www.volkswagen.co.uk/technology/comfort-and-convenience/gesture-control]
3. Charles J. Cohen, Glenn Beach, Gene Foulk: A Basic Hand Gesture Control System for PC Applications. Cybemet Systems Corporation (2001)
4. Shumilov, A., Philippovich, A.: Gesture-based animated CAPTCHA. Information and Computer Security, Vol. 24 Iss 3 pp. 242 - 254 (2015)
5. Dipietro, L., Angelo M. Sabatini, Dario, P.: urvey of Glove-Based Systems and their applications. IEEE Transactions on systems, Man and Cybernetics, Vol. 38, No. 4, pp 461-482 (2008)
6. Vicen, R., Garcia-Gonzalez, A.: Traffic Sign Classification by Image Preprocessing and Neural Networks. Conference: Proceedings of the 9th international work conference on Artificial neural networks (1970)
7. A.A. Vedenov, M.V. Zurin, E.B. Levchenko: Optical image preprocessing for neural network classifier system. In book: Parallel Problem Solving from Nature (2006)
8. Kussul, E., Baidyk, T., Kussul, M.: Neural network system for face recognition. Conference: Circuits and Systems, 2004. ISCAS '04. Proceedings of the 2004 International Symposium onVolume: 5 (2004)
9. Son Lam Phung, Bouzerdoum, A., Chai, D.: A novel skin color model in YCbCr color space and its application to human face detection. Visual Information Processing Research Group Edith Cowan University, Westem Australia, IEEE KIP, pp 289-292 (2002)
10. Brownlee, J.: What is the Difference Between Test and Validation Datasets? Machine Learning Process, [https://machinelearningmastery.com/difference-test-validation-datasets] (2017)
11. Ahmed El-Sawy, Mohamed Loey: CNN for Handwritten Arabic Digits Recognition Based on LeNet-5. In book: Proceedings of the International Conference on Advanced Intelligent Systems and Informatics (2016)
12. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. NIPS pp. 19 (2012)
13. Geoffrey E. Hinton, Nitish Srivastava, Krizhevsky A., Sutskever, I., Ruslan R. Salakhutdinov: Improving neural networks by preventing co-adaptation of feature detectors. Department of Computer Science, University of Toronto, pp. 1-18 (2012)