International Journal of Engineering Research and Technology. ISSN 0974-3154, Volume 13, Number 11 (2020), pp. 3611-3615
 https://dx.doi.org/10.37624/IJERT/13.11.2020.3611-3615

Development of Application for Recognition of Object Groups in the Image

Ilnur Saitovich Miftahov[1], Larisa Yurievna Grudtsyna[2], Irina Yurievna Myshkina[3]

[1]Student, Department of Information Technologies and Energy Systems, Naberezhnye Chelny Institute, KFU, Russia.
ORCID ID: 0000-0002-4121-6196, Kazan Federal University,
[2]Senior Lecturer, Department of Information Technologies and Energy Systems, Naberezhnye Chelny Institute, KFU, Russia.
ID Scopus: 57189270650, ORCID ID: 0000-0003-1307-6605, Kazan Federal University,
[3]Candidate of Technical Sciences, Associate Professor, Department of Information Technologies and Energy Systems, Naberezhnye Chelny Institute, KFU, Kazan Federal University, Russia. ID Scopus: 5718926839,
ORCID ID: 0000-0003-4309-3350,

Abstract

In this article, we solve the actual problem of recognizing the object group in the image. The solution to this problem resulted in an application developed on the basis of a neural network for automatic object recognition. We considered the problem of recognition of cars and trucks on images of any size. The developed application creates a neural network, allows it being trained, as well as using the trained network to solve the recognition problem. In this work, we created a convolutional neural network that allows detecting the features of cars under consideration and classifying them. The network architecture was selected in such a way that the result of its work was adequate. When selecting, we considered feed-forward neural networks, since they showed themselves in the best way in solving such problems. To select the structure and build this network, we studied the corresponding theoretical material about the main types of neural networks, as well as about the algorithms for their training. To train this network, we used an error backpropagation algorithm based on the gradient descent method when finding the minimum of the activation function. An important point is tracking of the network training results; to calculate the accuracy and error indicators (in time), the developed application creates the corresponding graphs. To recognize cars on an arbitrary image, it was divided into admissible parts, and then the image was transmitted in parts to the trained network. The network operation result is displayed using the graphical interface of the application. The objects "passenger car" and "truck" are localized and assigned to the corresponding class.

Keywords: convolutional neural network, feedforward network, fully connected layers, image subsampling, backpropagation algorithm.

I. INTRODUCTION

In recent decades, the world is rapidly developing the intelligent information systems (IIS) [1, 2]. They have found application in many spheres of social life and in many subject areas. Now, the IIS help to find the necessary information on the Internet, predict the weather, determine the state of a person from a photograph, assess the enterprise's personnel, regulate traffic, etc. [3-5]. These are exactly the tasks where it is necessary to process a large amount of data to achieve the final goal. And the result will largely depend on how accurately and quickly all the necessary data was analyzed. For this purpose, the ISS should be based on a computer with high computing capabilities and (most importantly) have a sufficiently large database of the required data.

One of the main IIS tools are neural networks (NN) and the current increase in interest in them is due to the fact that there are the necessary tools for their development and use now [6]. Namely, large data sets for analysis and accurate decisions, as well as the computing power of computers. Due to this, the artificial neural networks began to be used in many information systems.

Neural networks solve a large number of tasks: forecasting, approximation, recognition. At the moment, one of the most pressing problems is the problem of the so-called object recognition (or object detection) [7, 8]. Many companies and universities devote their resources to solve these problems and apply them in real life. For example, the car controlling IS, which is also based on a neural network that recognizes such objects as roads, cars, pedestrians, is now actively developed. However, there are various structures of neural networks that solve this problem and have different results. And it is not known in advance which of the structures will give the best result. Therefore, this direction requires constant improvement and refinement, the creation of new various structures to solve the assigned tasks.

II. METHODS

Let us consider the problem in the following setting. It is necessary to determine location of the "car" and "truck" objects groups in the image and carry out their classification. Input data are RGB color images of any size. The image can have the following formats: JPG, PNG. Output data are images with localized and classified "car" and "truck" objects.

To solve this problem, we created an application in C++ in Qt Creator. The main component of this application is a neural network that automatically recognizes objects belonging to the above groups. In the work, a convolutional neural network of direct propagation is being created; the network is trained and tested, and then used to recognize the object groups.

International Journal of Engineering Research and Technology. ISSN 0974-3154, Volume 13, Number 11 (2020), pp. 3611-3615
 https://dx.doi.org/10.37624/IJERT/13.11.2020.3611-3615

Let us consider a network created to solve the problem. This network has a number of convolutional, subsampled and fully connected layers. The chosen architecture is not random, the number and type of network layers are determined through practical tests and theoretical recommendations. For example, some previous network architectures had fewer feature maps in the convolutional and subsampled layers, which resulted in poor numeric results. The created network showed the best results among the tested networks (Fig. 1).

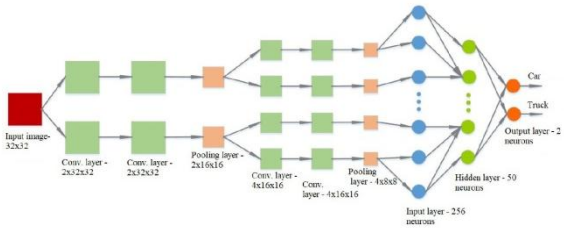

Fig 1. Network structure

An image of 32x32 pixels is transmitted to the created network, which can have either one color channel or three RGB color channels. This image is the input value. The first network layer is convolutional, with two feature maps, 32x32 pixel maps and 3x3 pixel filters. According to the recommendations, the number of feature cards should be taken one to two. If you increase the number of feature maps on the next layer, it is recommended to double the number. Two feature maps of the next layer are taken for one feature map of the previous layer.

The next layer is convolutional, with two feature maps and 32x32 pixel maps. The layer has 3x3 pixel filters. The first two convolutional layers are responsible for basic feature recognition. They serve to identify large object elements in the image. There is a subsampling layer with two feature maps and a filter size of 2x2 pixels after a cascade of two convolutional layers. This layer reduces the size of the feature map, thereby facilitating the network operation, since the number of subsequent neurons will be half as great. The next network layer will receive a map with dimensions of 16x16 pixels.

Then there is the convolutional layer with an increased number of feature maps after the subsample layer. This layer has 4 feature maps, map sizes - 16x16 pixels. Then there is another convolutional layer with 4 feature maps and map sizes also 16x16 pixels. These convolutional layers are responsible for small object features, such as line, bend, etc. Then there is the last subsampling layer with the same number of feature maps and filter sizes of 2x2 pixels after the next cascade of convolutional layers. This sub-sampling layer serves as the final downscaling before the fully connected layer.

Convolutional and subsampling layers are followed by a fully connected layer. This layer has an input layer of 256 neurons, one hidden layer of 50 neurons and an output layer of 2 neurons.

Since the task is to recognize two object types, this network has two output neurons, where each o them is responsible for its own class of object, respectively.

There are two activation functions used for these network neurons. In convolutional layers, the activation function is ReLu (x - input, $f(x)$ - output):

$$f(x) = \begin{cases} x, x \geq 0 \\ 0, x < 0. \end{cases}$$

The fully connected network layers use the Sigmoid function:

$$f(x) = \frac{1}{1 + e^{-x}.}$$

These function works well for recognition in deep learning, since it does not need complex mathematical calculations and has a simple derivative [7-9].

Network setup goes through two stages. The first stage is forward propagation, when the transmitted signal propagates from input to output neurons. After receiving an error, the signal propagates back and changes the network weights. This is called the backpropagation algorithm. The algorithm starts from the output neuron. The output neuron error is calculated by the formula [10]:

$$\delta_o = (OUT_{ideal} - OUT_{actual}) * \dot{f}(INP),$$

where OUT_{ideal} – desired neuron response, OUT_{actual} – received neuron output, $\dot{f}(INP)$ – derivative of the activation function with respect to the neuron input value.

Then these "deltas", errors on the output layer are passed to the hidden layers inside according to the formula [10]:

$$\delta_h = \dot{f}(INPUT) * \sum(w_i * \delta_i),$$

where $\dot{f}(INPUT)$ – derivative of the neuron activation function, for which "delta" is considered, w_i – neuron weight in the previous layer, δ_i – neuron error in the previous layer.

The next step is to update the neuron weights from the received error. A gradient is used to update the weights. The gradient is calculated by the formula [10]:

$$GRAD_B^A = \delta_B * OUT_A,$$

where A – point at the beginning of synapse, B – point at the end of synapse, δ_B – neuron weight, OUT_A – neuron output value.

After calculating the error, you need to update neurons on the layer using the formula [10]:

$$\Delta w_i = E * GRAD_w + \alpha * \Delta w_{i-1},$$

where E – training speed, α - training moment, Δw_{i-1} – weight changes in the previous training cycle.

Using the described algorithm, the entire multilayer network is processed from the end to the beginning, and the network neuron weights are updated.

To train the convolutional layer, "deltas" are also required, which are obtained during training by an error back propagation in a multilayer perceptron. Training for a convolutional layer is similar to training for a fully connected

International Journal of Engineering Research and Technology. ISSN 0974-3154, Volume 13, Number 11 (2020), pp. 3611-3615
 https://dx.doi.org/10.37624/IJERT/13.11.2020.3611-3615

layer with one difference - the layer is the neuron matrix that is trained in the same way.

To create an application that implements a neural network, we used an object-oriented programming approach [11, 12]. This approach is one of the best for solving this problem, since it is convenient and logically correct to represent the neural network elements as objects in a programming language. When developing the application, we applied the “clean architecture” approach [13]. We also used the "observer" pattern to better implement the program.

The main object classes can be divided into two groups, one group of classes is responsible for the algorithm operation, and the other - for the implementation of the application's interaction with the user [11]. The general structure of the application and all its classes are shown in Figure 2.

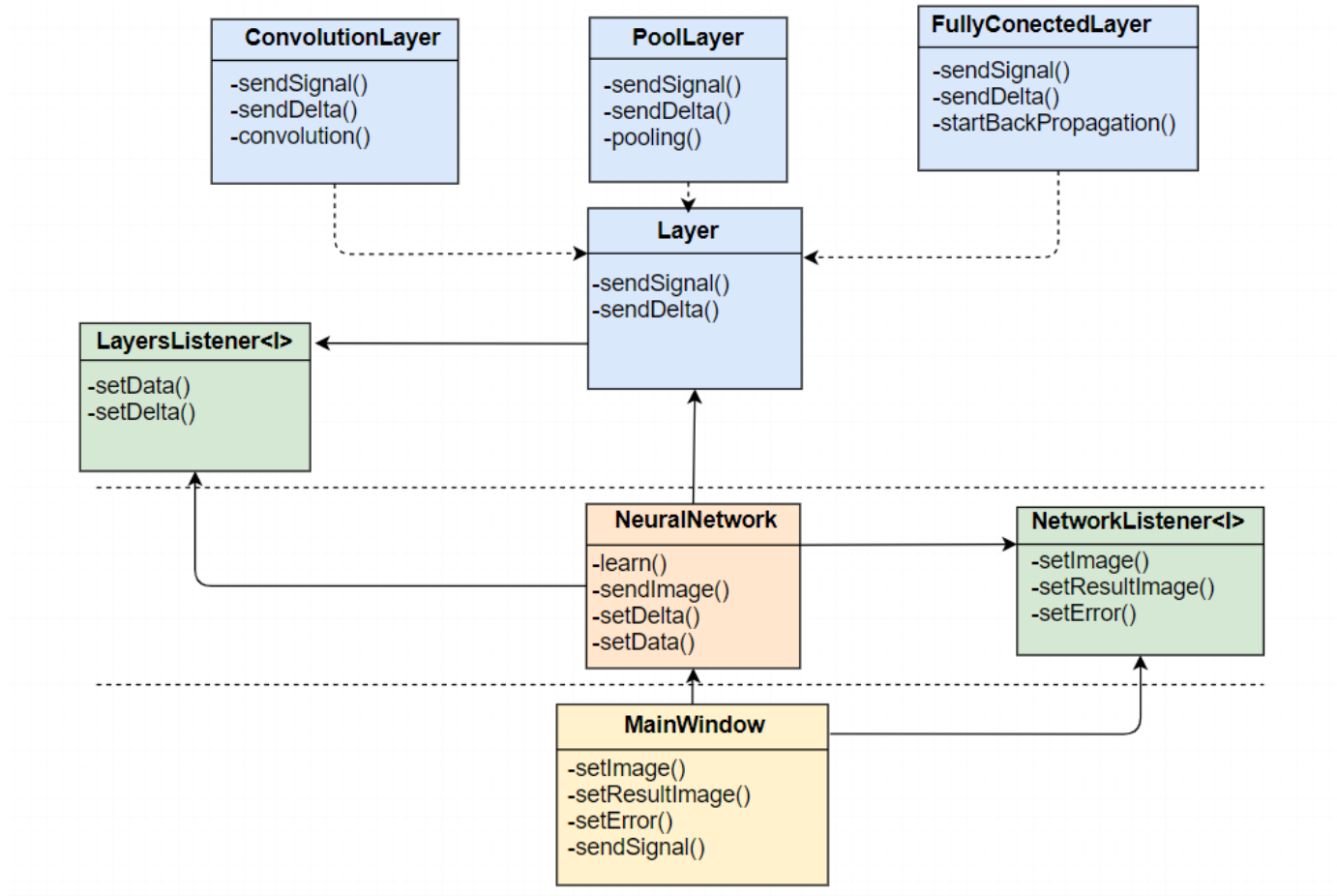

Fig 2. Application structure

The **MainWindow** class is responsible for the application interface, it implements the graphic window creation methods, as well as the user interaction methods.

The **NeuralNetwork** class is the class responsible for assembling and building a neural network. This class has a field in the form of a vector, into which various neuron layers are added. The class organizes the interaction between the layers and transforms the transmitted data.

The **Layer** class is the parent class for all kinds of layers in the network. Layer is an abstract class that only has pure virtual methods [8].

The **ConvolutionLayer** class is a child of the Layer class and is responsible for the operation of the convolutional layer of the neural network. The class accepts a signal in the form of a picture or a data vector, performs a convolution operation and passes the signal. On the return pass, the class receives the vector of errors, changes its kernels with the help of this data, as well as calculates the error on the previous layer and passes it to the next one.

The **PoolLayer** class is a child of the Layer class and is responsible for the subsampling operation. The class takes a data vector, reduces its dimension and passes it to the next layer. On the return pass, this class expands the resulting error vector using the inverse subsampling algorithm.

The **FullyConnectedLayer** class is a child of the Layer class. This class implements the functionality of a fully connected layer; it is a multilayer neural network. The class generates a network error and passes it to the previous layers. The accuracy of the neural network and the error are also formed in this class.

International Journal of Engineering Research and Technology. ISSN 0974-3154, Volume 13, Number 11 (2020), pp. 3611-3615
 https://dx.doi.org/10.37624/IJERT/13.11.2020.3611-3615

III. RESULTS AND DISCUSSION

To train and test the constructed neural network, we chose the Cifar-10 dataset. The Dataset used includes 6000 (5000 training and 1000 test) color images of 32x32 pixels of the "passenger car" class and a similar number of images of the "truck" class. It should be understood that this set has a small size, since much larger data sets are used to train networks and obtain high results.

Due to the application interface, it is possible to observe and evaluate the learning outcomes of the network. The values of accuracy and root mean square error, calculated during network training, are displayed on graphs. The average network error was 0.05% during training. The maximum accuracy value was 73%, which is a good result, taking into account the size of the training sample.

The application interface also allows uploading an image of arbitrary size to recognize cars of the specified groups on it. For clarity, a rectangular channel, proportional to those images that were used in the training set, will "slide" over the loaded image and transfer parts of the image to the neural network for further recognition. After that, all recognized areas in the image will be selected and classified (Fig. 3).

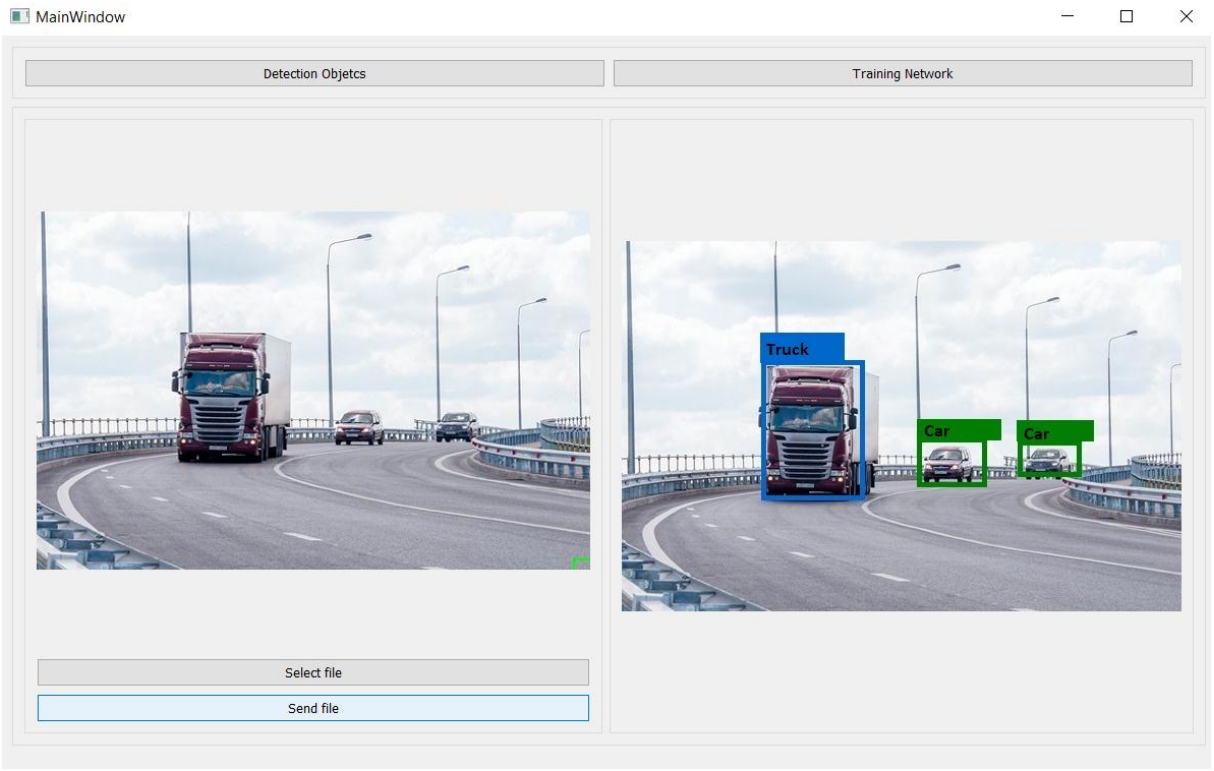

Fig 3. Application interface

IV. SUMMARY

We created a convolutional neural network for recognizing objects of the "truck" and "passenger car" classes in this work. The network includes a cascade of two convolutional layers, subsampling layer, repeated twice as the number of maps increases. There is a fully connected layer with two hidden layers at the end of the network. The Cifar-10 dataset was used to train this network. The convolutional network was trained on a 2.3 GHz Intel Core i5 processor. The training time was 150 epochs and 3.5 hours.

We also developed an application that allows: demonstrating the network learning process using graphs; loading an arbitrary image and visually evaluating the recognition result of objects belonging to the specified classes.

To check the application operation, we carried out some tests to recognize the "car" and "truck" objects, which gave a successful result in most examples. We can conclude that the constructed network works correctly and has good accuracy indicators. The application is stable and adequate.

V. CONCLUSIONS

The problem of recognizing objects in the image is one of the urgent problems in the theory of image recognition. If we talk about recognizing the "car" object group, then the solution of such problems is in demand for the autopilots, security cameras, and robotics at the present time.

We created an application in C++ to recognize trucks and cars in the image. During the development, we applied the principles of object-oriented programming, as well as the "clean architecture" approach, which allows creating easily scalable applications [14].

The developed application (with minor modifications) can be turned into an accessible library for creating neural network structures, including convolutional, fully connected and subsampling layers. This will allow using this program to solve a wider range of problems in the future. Further development of the application also involves the optimization of the algorithm and the use of this algorithm on video images, which is no less urgent task today.

ACKNOWLEDGEMENTS

The work is performed according to the Russian Government Program of Competitive Growth of Kazan Federal University.

REFERENCES

[1] Ostroukh AV, Nikolaev AB. Intelligent information systems and technologies. Monograph. St. Petersburg: Lan, 2019. 308 p.

[2] Blanchard BS, Fabrycky WJ, Fabrycky WJ. Systems Engineering and Analysis, Pearson. 2010;5: P. 800.

[3] Asanov AZ, Myshkina IYu. Investigation of the possibility of using neural networks in solving the problem of selecting a team for the project implementation, Management Issues. 2017;1:31-39.

[4] Myshkina IY, Asanov AZ, Grudtsyna LY. Evaluation and selection of personnel based on clear and fuzzy cognitive models. International Journal of Soft Computing. 2015;10:448-53.

[5] Badykov IV, Myshkina IYu, Grudtsyna LYu. Recognition of human states using the TENSORFLOW machine learning library, Scientific and Technical Bulletin of the Volga Region. 2019;5: 12-14.

[6] Khaikin S. Neural networks. Full course, M.: Williams. 2006: 1104 p.

[7] Kadurin A. Deep Learning, St. Petersburg: Piter. 2018: 480 p.

[8] Goodfellow I, Bengio Y. Courville A., Deep Learning, The MIT Press, 2016, P. 800.

[9] Tarikov R. Creating a neural network, M.: Williams. 2018: 272 p.

[10] Rojas R. Neural Networks: A Systematic Introduction, Springer. 1996: P. 552.

[11] Laforêt R. C ++, St. Petersburg: Piter. 2018: 982 p.

[12] Stroustrup D. C++ Programming Language, M.: Binom. 2018; 1136 p.

[13] Martin R. Pure Architecture. The Art of Software Development, St. Petersburg: Piter. 2018; 352 p.

[14] Stroustrup B. The C++ Programming Language: Special Edition, Addison-Wesley Professional. 2000; P. 1030.

Ilnur Saitovich Miftahov 4th year student in the specialty "Systems Analysis and Management" of the Naberezhnye Chelny Institute of the Kazan Federal University. He is a developer of client-server applications at the Scientific and Technical Center of KAMAZ PJSC. Research interests: development of artificial intelligence systems (AI), use of neural networks in AI. He plans to enter the magistracy.

Larisa Yurievna Grudtsyna She graduated from the Yelabuga State Pedagogical Institute. Senior Lecturer, Department of Systems Analysis and Informatics, Naberezhnye Chelny Institute, Kazan Federal University. Research interests: management in social and economic systems, intellectual management. The list of her scientific works includes more than 20 works.

Irina Yurievna Myshkina She graduated from the Yelabuga State Pedagogical Institute. Candidate of Technical Sciences, Associate Professor of the Department of System Analysis and Informatics, Naberezhnye Chelny Institute, Kazan Federal University. The list of her scientific works includes more than 30 works in the field of management in social and economic systems.