Designing architecture of an artificial neural network for classification using robotic complexes

K M Kononenko, D G Demidov, A I Vinokur and V L Krupenin
Department of Informatics and Information Technology, Moscow Polytechnic University, Moscow, ulitsa Pryanishnikova, 2A, 127550, Russia.

kononenkoxenia@gmail.com

Abstract. This article describes the basic principles of constructing artificial neural networks for pattern recognition on raster images using robotic complexes. The basic parts of convolutional neural networks are considered. Examples of evaluating the accuracy of preliminary testing results developed by the authors of a convolutional neural network model are given.

1. Introduction

Over the past decade, visual recognition has taken one of the leading positions in the development of machines using artificial intelligence. The use of visual information by automated production robotic complexes (automatic sorting, defining defects, determining the physicochemical properties of an object, etc.) has grown many times. From a technical point of view, for the information systems that form the basis of such complexes, visual information is still provided as a set of pixels. It is necessary to use and develop new processing methods for the robot to “consciously” process this information. One of these methods is the rapidly developing artificial neural networks, which are capable of teaching information systems to identify real-world objects, this teaching is like teaching a small child. Despite the obvious progress, this process remains slow and unpredictable.

Convolutional neural networks, which are most often used for visual recognition in industrial complexes, and have proven their effectiveness in comparison with classical fully-connected neural network architectures, also require improvement.

The challenge for engineers is to correctly design the architecture of the neural network, which would allow efficient recognition of images using robotic systems [1]. Currently, this task is extremely urgent both in pattern recognition by small robots and in mass production of goods – from food to mechanical engineering.

Let’s consider the possibility of designing a convolutional neural network for pattern recognition, in a particular case, classification (sorting by categories) of objects that can be implemented in a robotic complex.

The main reason of the study and creation of a new ANN model is the specification of the subject area (robotic complexes). If you look closely at the set of images of goods at a particular production, you can see that according to some visual features, the images will differ from the
set of the same images, for example, in an online store. This difference suggests that between
the patterns of goods made under the same conditions, there is some connection that you can
try to establish using the classifier. As a result, the model will have to produce useful predictions
on previously never seen data by combining the input data in a certain way – that is machine
learning [2].

In the context of this paper, pattern recognition means the ability of computer programs to
detect objects in raster images. Information systems for pattern recognition are built on the
example of the functions of the human brain, which is able to identify objects of the real world.
The brain of humans and animals easily recognizes objects, however computers encounter
considerable difficulties in performing this task [3]. Pattern recognition systems require deep
machine learning to achieve acceptable results in solving pattern recognition problems.

2. Convolution neural network
Most researchers agree that the convincing leader in pattern recognition productivity is the
convolutional neural network (CNN) [4]. Instead of pre-processing the image to obtain image
features such as textures and shapes, the CNN accepts only raw pixel image data as input and
“learns” to extract image features from them. Convolution operation is the basis of the
convolution neural network, that is using for pattern recognition.

In mathematics, convolution represents the operation of combining two functions to obtain
new values. In machine learning, this is a combination of a convolution filter (kernel) and an
input matrix of values. The convolution kernel (a matrix of weights) slides horizontally and
vertically along the matrix of pixel values of the image multiplying the matrix values
superimposed on each other, adds them and puts the resulting amount into the output feature
map (feature matrix).

During training, the CNN “remembers” the optimal values for the kernel, which allow to
extract significant elements (textures, edges, shapes) from the input map of objects. As the
number of filters (depth of the map of output features) applied to the input data increases, the
number of features that the CNN can extract increases along with it.

After each convolution operation, the ReLU activation function (Rectified Linear Unit,
formula 1) is applied to the output characteristic map [5]:

\[ F(x) = \max(0, x) \] (1)

For the CNN, the main advantage of the ReLU function is its speed and its ability to deal
with the exponential decay of the weights using the back propagation method because its
gradient at x > 0 is 1.

The next step is to subsample (pooling) feature maps. Subsampling allows to increase the
accuracy of image signs by selecting from the map of signs only part of the values. The
maximum function is the most used for this operation. With subsampling, the spatial volume of
feature maps is significantly reduced.

So, the structure of the CNN consists of repeating convolution and subsampling blocks. The
more blocks, the more features will be extracted from the image. One of the tasks of the
engineer designing such a network is to prevent an overabundance of signs and balance CNN
blocks in such way that recognition is as accurate as possible.

The final part of the multiclass SNS is the fully connected neural layers. All neurons of the
first layer connects to all the neurons of the second layer, creating the ability to transmit a signal
to each neuron of two layers. As a rule, the last fully connected layer contains the softmax activation function (formula 2):

$$\sigma(z_i) = e^{z_i} \left( \sum_{k=1}^{N} e^{z_k} \right)^{-1},$$  

(2)

where \((z_i)\) is the value at the output of the i-th neuron before activation, and \(N\) is the total number of neurons in the layer), which displays the probability value from 0 to 1 for each class that the model is trying to determine.

3. Architecture of CNN for pattern recognition

When designing the CNN, the researcher’s task is to correctly configure the model: choose the number of convolution blocks, convolution filters, setting up activation functions, etc. The selection of all parameters is individual in each subject area. Our CNN model for the recognition system is presented in figure 1.

![Figure 1. The structure of the developed CNN](image)

The developed CNN is used in two modes: administration (training) and use (network makes a prediction).

The input data in training mode is a set of labeled images, in use mode - only one image of unknown class. CNN accepts the input pixel values with a depth of 1 (we give a black and white image).

The size of the convolution kernel is 3x3. The small size of the filter allows you to extract more details, but it is very sensitive to noise.

The size of the subsampling matrix is defined as 2x2. This size is suitable for our model, because the input image is small (4096 pixels), if we increase the size of the input image during testing, then the size of the subsampling matrix should also be increased.

The final layer of the network is fully connected layer, where the number of neurons is equal to the number of predicted classes. During the operation of the CNN, the output gives the probability for each class. So, the greatest value of probability can be understood as determining whether an object in an image belongs to a corresponding image.

4. Evaluation of the result

Neural network training is the most important stage in preparing an ANN for solving assigned tasks. When using ANN for classification, the learning process occurs with the teacher. This
means the presence of a training set that contains examples with already classified values: pairs of values for class names and images. The task of training is to select a function (data function) that would be able to correctly distinguish between classes. The selection of the data function occurs by adjusting the weights of the ANN with repeated runs of the training set. To assess the quality of the adjustment, and therefore the training, a loss function is used, which describes the difference between the obtained and the desired results and allows you to quantify the quality of the selected weights. Intuitively, the lower the value of the loss function, the better the network is configured, and the higher the value, the worse predictions [6].

We track the values of the loss function at each training step (figure 2) to control the learning process (network learning or not). This allows us not to validate the network after each step.

According to the graph, there are oscillations of the loss function in the first steps, that is, the algorithm has low gradient convergence.

An important observation in analyzing the performance of the CNN model for multiclass classification is that the operated data set in the network is unbalanced. Therefore, when evaluating effectiveness, methods such as accuracy or mathematical correlation are not reliable. The F1-score is used to evaluate an unbalanced data set.

The F-score is determined by finding the metrics Precision (accuracy) and Recall (completeness), which are used in the assessment of the majority of data mining algorithms.

Table 1 shows the calculations F-score for all the evaluated models (Model No. 1 – 6000, Model No. 2 – 3000, Model No. 3 – 1000, Model No. 4 – 500 trainings respectively).

| Model name   | Precision | Recall | F-score |
|--------------|-----------|--------|---------|
| Model No.1   | 0.431     | 0.450  | 0.440   |
| Model No. 2  | 0.400     | 0.400  | 0.400   |
| Model No. 3  | 0.320     | 0.323  | 0.321   |
| Model No. 4  | 0.363     | 0.373  | 0.367   |

According to table 1, the most accurate model is a model with a lot of training. Thus, previously it can be concluded that the network structure is designed correctly, but for more accurate results requires more training.
5. Conclusion
The success of recognition systems largely depends on how well the system is trained. Obviously, more than half of the success depends on the structure used by the CNN. The increased use of AI by robotic complexes for pattern recognition leads to a revision of the classical structures of neural networks – neural networks require a special approach to design, taking into account all aspects of the system.

In this paper, a solution to the problem of pattern recognition for robotic systems was proposed. The accuracy of this model was 44%, but due to the experiments, you can see a tendency to increase recognition accuracy due to the improvement of the learning process (increase in the number of samples and training).

References
[1] Hao-Tien Lewis Chiang, Aleksandra Faust, Lydia Tapia 2018 Deep Neural Networks for Swept Volume Prediction Between Configurations [Electronic resource.] Access mode: https://ai.google/research/pubs/pub46943
[2] Mestetsky L 2011 Mathematical methods for pattern recognition. [Electronic resource.] Access mode: https://www.intuit.ru/studies/courses/2265/243/info
[3] Stanford University 2019 Neural Networks. [Electronic resource.] Access mode: http://cs231n.github.io/neural/
[4] Stanford University 2019 Convolutional Neural Networks (CNNs / ConvNets). [Electronic resource.] Access mode: http://cs231n.github.io/convolutional-networks/ - architectures
[5] Milutin I 2018 Neural network activation functions: sigmoid, linear, stepwise, ReLu, tanh. [Electronic resource.] Access mode: https://neurohive.io/ru/osnovy-data-science/activation-functions/
[6] Hao Zhang 2017 Loss Functions in Neural Networks. [Electronic resource.] Access mode: https://isaacchanghau.github.io/post/loss_functions/