FEATURES OF NEURAL NETWORK APPLICATION IN AUTOMATION PROCESSES

Annotation. The process of selecting a convolutional neural network for the RaspberryPi microcomputer, which is part of a sorting machine – a device for collecting and sorting plastic bottles and aluminum cans, is shown. Three neural networks were selected for training: AlexNet, SqueezeNet and MobileNet. The training was conducted using Transfer Learning in two ways: replacing the last level of the classifier, then training and configuring the entire network. The Caffe framework was chosen as the basis for launching neural networks as the most popular. For the verification sample consisting of 2300 photos, which includes photos of cans, bottles and "other garbage", the correct recognition results are given, and the verification result is described in more detail in a small trial sample of 30 photos. The work gives time characteristics of neural networks on a computer and RaspberryPi. Experiments show that the MobileNet network has the highest percentage of correct recognition, while RaspberryPi has a network where the minimum time required to process one image belongs to SqueezeNet.

Keywords. Neural network, automation, networks, sorting, deep learning.

Introduction. The problem of waste disposal is currently particularly relevant. To dispose of waste, it is necessary to sort them before recycling. General garbage can be sorted during and after the collection period as assemblies.  

Pre-sorting sorting machine-a device for collecting plastic bottles and aluminum cans, as well as during the garbage collection period, it is recommended to reject (not collect) "other garbage" to be sorted. The sorting machine consists of a microcomputer (Raspberry Pi) to which a video camera is connected, sorting drives and two containers in which there are bottles and cans. Sorting is determining whether the item is a plastic bottle, an aluminum can, or both using a photo of an item placed in a sorting machine. The sorting problem is that an object on a device with a limited number of resources needs to be classified within one second. To do this, it is recommended to use a deeply trained convolution network using the Caffe software planform (framework). The paper presents a comparison of the quality of recognition for three networks: AlexNet, SqueezeNet, MobileNet.

Reasons for choosing convolutional neural networks. Convolutional networks are a type of neural networks consisting of filters (svertki). Each such filter learns to distinguish information from different lines and loops (first layers) to specific images (last layers) through neural network training [2].

The first convolutional neural networks were invented by Y. Lekun in 1989 and used to recognize handwriting characters [3]. Such a network is called LeNet. However, then neural
networks were not widely used, since there was not enough computing power for the fast operation of networks.

In 2012, Alex Krizhevsky's team returned to convolutional neural networks when he took first place in the ILSVRC-2012 competition [4], whose decision in terms of correct recognition was significantly ahead of the best decision of 2011 (ILSVRC-2011). Alex Krizhevsky's convolutional network is called AlexNet [5] and is very similar to LeNet.

In the future, on the basis of the AlexNet Network, Network-in-Network (NiN) [6] was created, in which small filters of 1×1 size began to be used, and the network itself consists of several networks, which increased the accuracy of recognition. Further development consists in the creation of a VGG [7] network, which also consists of small filters (3 × 3 and 1 × 1) and more layers. In the future, the idea of increasing the number of layers was developed by the Google Group, which created the GoogLeNet network [8]. This network also uses 3 × 3 and 1 × 1 small filters, and based on the idea of the NIN network, the GoogLeNet network consists of sequential small networks (modules) called Inception.

Such deep networks have even become a separate area of machine learning called" [9]. These networks show a high degree of correct recognition, but their speed of operation is very low. In connection with the development of mobile technologies and computing in mobile devices (smartphones), the developers of neural networks wanted to transfer their neural networks to smartphones. Therefore, the SqueezeNet [10] and MobileNet [11] networks were created. SqueezeNet has many layers consisting of small convolutions, but the number of parameters is several times smaller than AlexNet (and therefore the number of calculations is smaller), the correct classification in percentages is the same as AlexNet. MobileNet, like Squeeze Net, has many layers, but the number of settings is even less than SqueezeNet, and on smartphones this neural network works faster.

The article examines the correctness of recognizing cans and bottles through the AlexNet, SqueezeNet and MobileNet networks, since the speed of work in RaspberryPi in these networks satisfies the given task (processing time less than 1 s).

**Materials and methods.**

The process of learning neural networks. Network training consisted of 384 cores at 993 MHz and 2048 MB of graphics memory on the NVIDIA GeForce GT 740m graphics card. For training, Caffe was chosen—a popular structure with a multi-worker community [12] and the model is a learning result that can be easily transferred from one computer to another and run on the CPU, even if the training is conducted on the GPU. To run models in RaspberryPi, the dnn module from OpenCV is used. OpenCV is built using NEON and VFPV3, which increases the speed of the neural network in RaspberryPi. The pre - prepared line was tested on a computer with an 8-core Intel Core i7-3610qm central processor, a 2.3 GHz base clock, and 8 GB of RAM. The prepared network was tested on RaspberryPi 3 with a Broadcom BCM4 central processor with 2837 cores, a frequency of 1.2 GHz and 1 GB of RAM. The training was done on 500 photos of bottles and 500 photos of cans downloaded from Google search queries, in addition, 10,000 images from the UKBench database were used as images of "other garbage" [13]. Bottles and jars of the training model are placed mainly vertically, with the neck facing up, and are not wrinkled. During the study, the check is carried out on a sample of the check, consisting of 150 photos of bottles, as many cans and 2000 "other garbage". This sample consists of photos that are not in the training sample. A small sample for testing consists of a photo of 10 bottles, jars of the same size and "other garbage". This sample contains photographs of bottles and cans, the neck is vertically down and up, as well as crumpled and unbroken jars and bottles are included.
For training, "educational" technology was used (Transfer Learning) [14]. Its essence is that neural networks learn not from "zero", that is, not from random weight values, but from an already prepared model. Models can be trained in both ImageNet [15] photo sets and other photo sets. Without the use of Transfer Learning technology, the training of neural networks would have to be on a much larger data set, and such training would take several weeks. AlexNet, SqueezeNet trained models are derived from Model Zoo [16]; MobileNet – [17].

Transfer Learning is divided into two types.

1. using a convolutional network to obtain symbols. All weights of the trained network are transferred to the new network and remain the same, but the last layer (classifier or classifier) is removed and a new classifier is created, which is taught "from scratch" in the new data set.

2. fine tuning of the convolutional network (nastroika(fine tuning). All weights of the trained network are transferred to the new network and adjusted taking into account the new data set, that is, the training is not made from a random State, but from a network that can distinguish the objects of any data set.

Next, both learning options are considered for all three convolutional networks.

### Results and Discussions

The results of the experiment. In Caffe, the learning process is determined by the number of iterations. The number of iterations is the number of parts of the training model (batch) in which the training was conducted [18]. Due to the memory limitations of the video card, parts of different sizes were used to train different networks (the larger this size, the more memory is required). In our opinion, the number of long periods of time is more pronounced, which indicates how many complete training samples have passed in the learning process. The size of the sample part affects the time it takes for the neural network to process the sample (the larger this size, the more time), the distribution of correct recognition (the smaller this size (as a percentage), the larger the distribution, the larger this size, the more correct recognition will gradually increase (as a percentage)), and also the neural network may not read.

The dimensions of the parts used for training on each network are indicated (Table 1). In the column "one floor", only the network in which the classifier is trained is marked. In the "whole network " column, a network is marked in which all layers are trained.

<table>
<thead>
<tr>
<th>Networks</th>
<th>Dimensions of parts</th>
<th>Iterations</th>
<th>Correct recognition, %</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>One layer</td>
<td>The entire network</td>
<td>One layer</td>
</tr>
<tr>
<td>AlexNet</td>
<td>64</td>
<td>64</td>
<td>4000</td>
</tr>
<tr>
<td>SqueezeNet</td>
<td>32</td>
<td>32</td>
<td>1000</td>
</tr>
<tr>
<td>MobileNet</td>
<td>8</td>
<td>8</td>
<td>2000</td>
</tr>
</tbody>
</table>

To test the network, samples were taken that reached the maximum recognition accuracy for the test sample. This maximum is achieved in different ways for different neural networks, but (based on experience) after the 1000th iteration, the correct recognition practically does not change, so the number of training iterations for all models is determined by 1000. When fine-tuning the entire network, AlexNet provides a graph of changing the correctness of recognition to test the neural network sample (Figure 1.a).
As shown in the figure, correct recognition in AlexNet reached about 98.2% in 1000 iterations. Further iterations change the correct recognition within 0.1%.

When fine-tuning the entire network, a graph of changing the correctness of recognition is shown for testing a sample of the SqueezeNet neural network (Figure 1.b).

Correct recognition in SqueezeNet reached about 97.6% in 1000 iterations, then the 2000th iteration increased correct recognition by about 0.8%, and the 3000th iteration by about 0.2%. Further iterations change the correct recognition within 0.1%.

Shows a graph of changing the correctness of recognition for testing a sample of the MobileNet neural network during fine-tuning of the entire network (Figure 1.c).

Correct recognition in MobileNet reached about 97.7% in 1000 iterations, then the 2000th iteration reduced correct recognition by about 0.05%, the 3000th increased by about 0.5%; the 4000th and 5000th iterations barely changed correct recognition, the 6000th increased correct recognition by about 0.05% 0.15%. Further iterations change the correct recognition within 0.1%.

Table 2

<table>
<thead>
<tr>
<th>Setting</th>
<th>AlexNet</th>
<th>SqueezeNet</th>
<th>MobileNet</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>One layer</td>
<td>The entire network</td>
<td>One layer</td>
</tr>
<tr>
<td>sample 1</td>
<td>1 100</td>
<td>1 100</td>
<td>1 100</td>
</tr>
<tr>
<td>sample 2</td>
<td>1 47</td>
<td>1 59</td>
<td>1 77</td>
</tr>
<tr>
<td>sample 3</td>
<td>1 100</td>
<td>1 100</td>
<td>1 100</td>
</tr>
<tr>
<td>sample 4</td>
<td>1 99</td>
<td>1 100</td>
<td>1 77</td>
</tr>
<tr>
<td>sample 5</td>
<td>0 57</td>
<td>0 100</td>
<td>0 49</td>
</tr>
<tr>
<td>sample 6</td>
<td>1 100</td>
<td>1 100</td>
<td>1 100</td>
</tr>
<tr>
<td>sample 7</td>
<td>0 100</td>
<td>0 100</td>
<td>0 100</td>
</tr>
<tr>
<td>sample 8</td>
<td>1 98</td>
<td>1 100</td>
<td>0 74</td>
</tr>
</tbody>
</table>
Specifies the maximum correct recognition for each network for the verification Sample. The maximum correct recognition for the verification sample is achieved in the SqueezeNet network with all prepared layers and is 98.64%.

Displays recognition results across each network for a small test sample (Table 3). Whether the neural network recognized the pattern correctly (1 is correct, 0 is incorrect). the column " % " indicates the degree of reliability of the neural network as to which class the sample is attributed.
Table 3

<table>
<thead>
<tr>
<th>sample</th>
<th>Neck up</th>
<th>Crumpled</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>+</td>
<td>–</td>
</tr>
<tr>
<td>2</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>3</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>4</td>
<td>–</td>
<td>+</td>
</tr>
<tr>
<td>5</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>6</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>7</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>8</td>
<td>+</td>
<td>–</td>
</tr>
<tr>
<td>9</td>
<td>+</td>
<td>–</td>
</tr>
<tr>
<td>10</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>11</td>
<td>+</td>
<td>–</td>
</tr>
<tr>
<td>12</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>13</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>14</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>15</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>16</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>17</td>
<td>+</td>
<td>–</td>
</tr>
<tr>
<td>18</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>19</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>20</td>
<td>+</td>
<td>–</td>
</tr>
</tbody>
</table>

If the sample has the property specified in the column name, then "+" is put in the cell, if it is not, then "−". Not a single network was able to recognize models 5 and 7—an inverted crumpled plastic bottle. In the op computer and RaspberryPi, the time of recognition of one image for each network is indicated (Table 4).

Table 4

<table>
<thead>
<tr>
<th>Networks</th>
<th>Time / picture on the central processor, Ms</th>
<th>Time / picture in RaspberryPi / image, Ms</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>One layer</td>
<td>The entire network</td>
</tr>
<tr>
<td>AlexNet</td>
<td>174</td>
<td>181</td>
</tr>
<tr>
<td>SqueezeNet</td>
<td>373</td>
<td>400</td>
</tr>
<tr>
<td>MobileNet</td>
<td>118</td>
<td>117</td>
</tr>
</tbody>
</table>

The recognition time of AlexNet and MobileNet in RaspberryPi has increased several times, and SqueezeNet has decreased. This is primarily due to the good optimization of OpenCV for ARM processors and the low convolutional outputs of SqueezeNet (SqueezeNet has a maximum output of 256, MobileNet-1024, AlexNet-384, and AlexNet also has two fully connected layers with 4096 outputs, which can also increase image processing time).
Thus, the MobileNet network is the highest when it comes to recognizing all layers correctly among those shown in Table 4. Table 6 between the times shown, the minimum time for processing a single image on a RaspberryPi is in the SqueezeNet network.

**Conclusions.**

In research work, RaspberryPi, which is part of the sorting automaton, provides a convolutional neural network selection process for its microcomputer. The AlexNet, SqueezeNet, and MobileNet neural networks were compared with two training options: classifier-only training and entire network setup. When comparing the correct classification of all networks for a small test sample, the MobileNet network is well recognized when recognizing all layers. All networks considered in this article are launched in RaspberryPi. As a result, the operating time of all networks, except SqueezeNet, has increased several times. Further research may go into the following areas:

1) Improve the architecture of the MobileNet neural network in order to increase correct recognition (as a percentage).
2) Creating a neuron capable of achieving more correct recognition (as a percentage) in a modified neural network.
3) Increase the training model.
4) Enable image pre-processing.
5) Find a solution for RaspberryPi that will save time on image processing ways of neural networks.

**REFERENCES**


DL4J, Torch7, Theano and Caffe. URL: https://deeplearning4j.org/comparing-dl4j-tensorflow-pytorch

UKBench Dataset. URL: https://archive.org/details/ukbench

Transfer Learning. URL: http://cs231n.github.io/transfer-learning/


Model Zoo. URL: https://github.com/BVLC/caffe/wiki/Model-Zoo

MobileNet-Caffe. URL: https://github.com/shicai/MobileNet-Caffe


Aiysbek Zhussip, master's student, Almaty Technological University, Almaty, Kazakhstan, ayisbek@mail.ru

Gulzhan Kashaganova, PhD, Almaty Technological University, Turan University, Almaty, Kazakhstan, guljan_k70@mail.ru

Kulzhan Togzhanova, PhD, docent, Almaty Technological University, Almaty, Kazakhstan, togzhanova_kuljan@mail.ru