Artificial neural network technology for text recognition

Anna V. Pyataeva[1,2], Sergey A. Genza[1]

[1] Siberian Federal University, Krasnoyarsk, Russia, anna4u@list.ru
[2] Reshetnev Siberian State University of Science and Technology, Krasnoyarsk, Russia

Abstract. The paper shows the use of artificial neural networks for the task of scene text recognition. A comparative analysis of the effectiveness of the use of various neural network architectures was presented. Experimental studies were performed on the MNIST, IIIT 5K, SVT and Synth 90k datasets.

Keywords: convolutional neural network, text recognition, MNIST dataset.

1 Introduction

Nowadays, a lot of work in the framework of computer technology based on deep learning methods is devoted to the subject of text recognition [1-4]. Irregular text is widely used. However, it is considerably difficult to recognize because of its various shapes and distorted patterns. [5]. Examples of scene text can be photographs of street signs, stills from movies with subtitles, data in robot navigation systems, etc. Among deep learning methods, convolutional neural networks (CNN) are the most studied [6-11]. Often, a popular MNIST dataset, which is a database of handwritten text samples, is used to conduct experimental research in text detection works. In [12], modifications of convolutional neural networks were studied to improve the accuracy of the classification of handwritten numbers. Authors [13] seek to characterize the learning architectures exploited in biological neural networks for training on very few samples, and port these algorithmic structures to a machine learning context. They carried out modeling of the structure of the neural network of smell of moths, then the created computational models are taught to read handwritten numbers. Most methods based on convolutional neural networks extract image features at the last level of the network using a single CNN architecture with an unlimited number of quantization approaches. What limits the use of intermediate layers to identify local features of the image.

2 CNN development for text recognition

Convolutional neural networks are part of deep learning technology, which has been the subject of many scientific papers in the last decade. Deep learning is a set of machine learning methods based on feature learning rather than specialized algorithms used to solve specific problems. Deep learning technology is actively used in computer vision, machine translation, speech recognition, and the quality of problem solving in these areas with the use of deep learning neural networks exceeds human efficiency. CNN is a special architecture of artificial neural networks, proposed by Yann LeCun [14, 15] aimed at effective image recognition, is a part of deep learning technologies. CNN's idea is to alternate between convolution layers and subsampling layers. CNN can have separable weights, that is, some neurons of some layer can use the same weights. Neurons using the same weights are combined into feature maps, and each neuron of such a map is associated with a part of the neurons of the previous layer. When calculating the output of the CNN, it turns out that each neuron performs a convolution of a region of the previous layer. Convolution is a mathematical operation applied to two functions f and g, generating a third function, which can be considered as a modified version of one of the original ones. It is a special kind of integral transformation. Consider the mathematical formulation of the convolution operation. Let f and g: $R^d \to R$ be two functions integrable with respect to the Lebesgue measure on the space R^d. Then their convolution is the function $f \otimes g : R^d \to R$, which is determined by the formula:

$$(f \otimes g)(x) \stackrel{def}{=} \int_{R^d} f(y)\, g(x-y)dy = \int_{R^d} f(x-y)\, g(y)dy. \quad (1)$$

From an image analysis perspective, a convolution operation is the operation of calculating the new value of a selected pixel in an image, taking into account the values of the surrounding pixels. A matrix called the convolution kernel is used to calculate the result of the convolution operation. The convolution kernel is usually represented by a square matrix n×n, n is an odd number. During the calculation of the new value of the selected pixel, the convolution kernel is applied by its center to this pixel. The surrounding pixels are also covered by the kernel. Next, the sum is

calculated, where the summands are the product of the pixel values on the values of the kernel cell that covered the pixel. The sum is divided by the sum of all elements of the convolution kernel. The resulting value is the new value of the selected pixel. By applying a convolution operation to each pixel of the image produces an effect that depends on the selected convolution kernel. In this work, we have constructed CNN for the task of recognizing handwriting in Arabic numerals and the scene text recognition.

2.1 Text recognition by MNIST

Neural networks of the following architectures have been developed to recognize handwritten digits.

2.1.1 Architecture 1. Neural network with two full connection layers

- First fully connected layer, ReLU activation function, 512 neurons;
- Second fully connected layer, Softmax activation function, 10 neurons;
- Cross-entropy loss function.

2.1.2 Architecture 2. Neural network with three full connection layers

- First fully connected layer, ReLU activation function, 1024 neurons;
- Second fully connected layer, ReLU activation function, 512 neurons;
- Third fully connected layer, Softmax activation function, 10 neurons;
- Cross-entropy loss function.

2.1.4 Architecture 3. Convolutional Neural Network with two Dropout Layers

- First convolution layer, ReLU activation function, convolution kernel 5x5, (28, 28, 32);
- Second convolution layer, ReLU activation function, convolution kernel 5x5, (28, 28, 32);
- First max-pooling layer, kernel 2x2, step 2x2, (14, 14, 32);
- First Dropout layer, neurons dropout probability is 0.5;
- Third convolution layer, ReLU activation function, convolution kernel 3x3, (14, 14, 64);
- Fourth convolution layer, ReLU activation function, convolution kernel 3x3, (14, 14, 64);
- Second max-pooling layer, kernel 2x2, step 2x2, (7, 7, 64);
- Second Dropout layer, neurons dropout probability is 0.5;
- First fully connected layer, ReLU activation function, 256 neurons;
- Second fully connected layer, Softmax activation function, 10 neurons;
- Cross-entropy loss function.

2.1.4 Architecture 3. Convolutional neural network with thee Batch Normalization layers

- First convolution layer, ReLU activation function, convolution kernel 5x5, (28, 28, 32);
- Second convolution layer, ReLU activation function, convolution kernel 5x5, (28, 28, 32);
- First Batch Normalization layer;
- First max-pooling layer, kernel 2x2, step 2x2, (14, 14, 32);
- Third convolution layer, ReLU activation function, convolution kernel 3x3, (14, 14, 64);
- Fourth convolution layer, ReLU activation function, convolution kernel 3x3, (14, 14, 64);
- Second Batch Normalization layer;
- Second max-pooling layer, kernel 2x2, step 2x2, (7, 7, 64);
- First fully connected layer, ReLU activation function, 256 neurons;
- Third Batch Normalization layer;
- Second fully connected layer, Softmax activation function, 10 neurons;
- Cross-entropy loss function.

The input gray-scale image of size 28x28 pixels (for neural networks without the convolutional layers have to be reduced to the vector of size 784) were subjected to normalization by aligning interval of the brightness values of the pixels to the range from 0 to 1. This image was sent to the input of the neural network along with the corresponding class label. The class label was converted into a unitary code (one-hot encoding) before being fed to the network input for the correct operation of the Softmax function and the cross-entropy function. Adam was chosen as the optimizer with a training step of 0.001, the batch size, for which a one-time adjustment of the scales is 128. The network weights before training are initialized to normally distributed values with a mathematical expectation of 0 and a standard deviation of 0.1. The bias of the layers is initialized to zero. For extreme pixels in convolutional layers, the values are automatically filled with zeros to obtain the same dimension at the output of the layer as at the input.

2.2 Scene text recognition

To solve the problem of scene text recognition, it is proposed to use a combination of a convolutional neural network, a recurrent neural network (RNN) [16, 17], and Connectionist Temporal Classification (CTC-loss) [3]. Feature extraction using a convolutional neural network is shown in Figure 1.

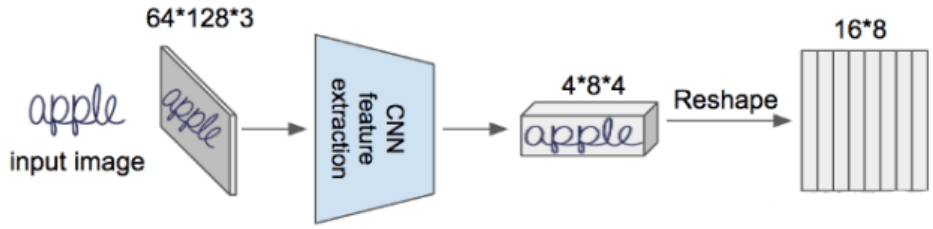


Figure 1. Feature extraction using a convolutional neural network.

The output of the convolutional network is a three-dimensional array that must be converted to a two-dimensional array, the second dimension of which must be equal to the maximum number of characters in the input image. After conversion, each column of the matrix is fed to the input of the corresponding LSTM-cell. The recurrent layer is used to represent the input features as a sequence and the network recognizes the characters exactly in the order in which they appear in the search text. Each vector is transferred to a fully connected layer and to the Softmax activation function. The dimension of a fully connected layer is equal to the length of the array of all possible recognizable characters with the addition of a blank marker. This marker CTC-loss algorithm uses to indicate the absence of a character in the text, because most of all input sequences of characters will be less in length than the maximum possible. The softmax function represents the input vector as a probability distribution vector for all possible symbols, including a blank symbol marker. The principle of the recurrent layer is shown in Figure 2.

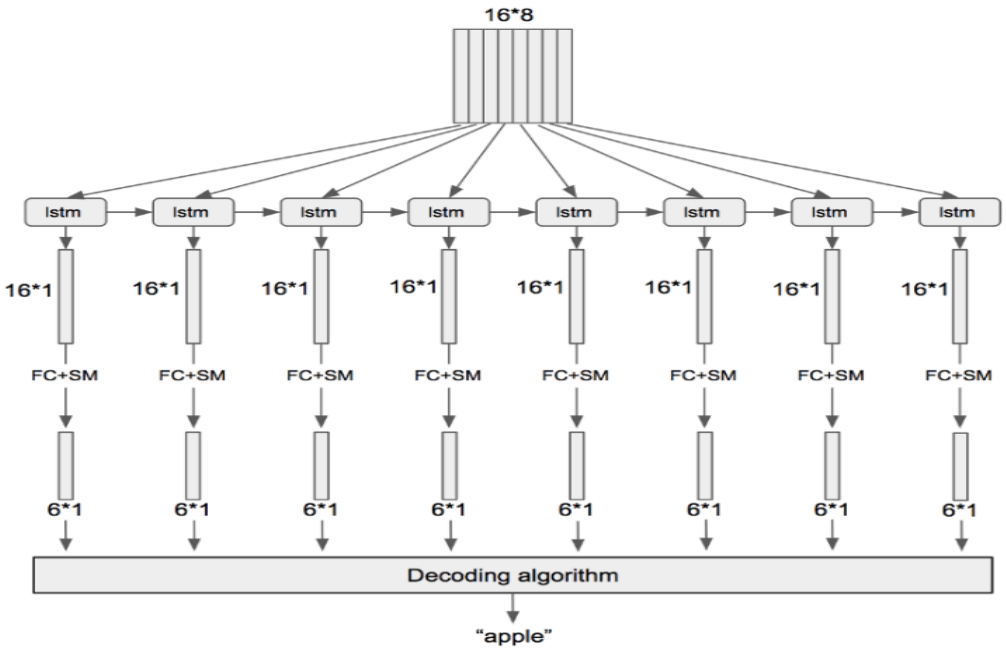


Figure 2. The principle of the recurrent layer.

Thus, the output is a matrix with distributed Softmax function by probability columns, where the number of columns is the maximum possible word length, and the number of rows is the length of the "alphabet" L + 1 (blank-marker). This matrix is input to a decoding algorithm that removes all duplicate characters, and then blank markers. The final alphabet is the set of all possible characters in a word and a blank marker by Eq.2:

$$L' = L \cup \{blank\}. \tag{2}$$

The probability of the path Y is equal to the product of the probabilities of all activated cells of each column of the matrix:

$$p(Y) = \prod_{t=1}^{T} y_{Y_t}^t \,,\ \forall Y \in L^{'}. \tag{2}$$

Example of calculating the probability of the path “ap-pl-ee”:

$$p(\text{"ap}-\text{pl}-\text{le)"} = y_a^1 \cdot y_p^2 \cdot y_-^3 \cdot y_p^4 \cdot y_l^5 \cdot y_-^6 \cdot y_e^7 \cdot y_e^8 \ . \tag{3}$$

The decoding algorithm is shown in Figure 3. Many paths always correspond to one word, and to calculate the probability of a word, it is necessary to calculate the sum of the probabilities for all possible paths.

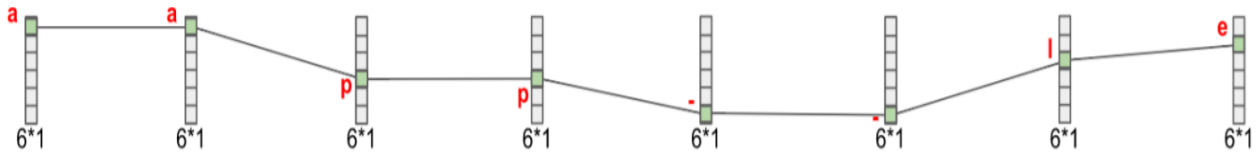


Figure 3. The decoding algorithm illustration.

The error function is similar to binary cross-entropy, only the word probability is used here as the class probability by Eq 4.

$$CTCloss = -\ln(p(\text{"}apple\text{"})) \,. \tag{4}$$

For the considered example, in which the alphabet consists of 6 characters, and the maximum number of characters is 8, the number of possible paths is 68 = 1679616 possible paths. When solving practical problems, the length of the alphabet and the maximum number of characters are significantly larger. Therefore, dynamic programming is used to calculate the word probability. The word probability value is used in the back propagation algorithm.

2.2.1 Architecture 5. Convolutional Recurrent Neural Network with CTC loss

For scene text recognition developed a neural network combining convolutional, recurrent layers and CTC error function (CRNN). Details of the CRNN architecture are given below.

- First convolution layer, ReLU activation function, convolution kernel 3x3, (31, 100, 64);
- First max-pooling layer, kernel 2x2, step 2x2, (16, 50, 64);
- Second convolution layer, ReLU activation function, convolution kernel 3x3, (16, 50, 128);
- Second max-pooling layer, kernel 2x2, step 2x2, (8, 25, 128);
- Third convolution layer, ReLU activation function, convolution kernel 3x3, (8, 25, 256);
- First Batch Normalization layer;
- Fourth convolution layer, ReLU activation function, convolution kernel 3x3, (8, 25, 256);
- Third max-pooling layer, kernel 2x2, step 2x2, (8, 13, 256);
- Fifth convolution layer, ReLU activation function, convolution kernel 3x3, (8, 13, 512);
- Second Batch Normalization layer;
- Fifth convolution layer, ReLU activation function, convolution kernel 3x3, (8, 13, 512);
- Sixth convolution layer, ReLU activation function, convolution kernel 3x3, (8, 13, 512);
- Fourth max-pooling layer, kernel 2x2, step 1x2, (8, 7, 512);
- Seventh convolution layer, ReLU activation function, convolution kernel 3x3, (8, 7, 512);
- Reshape- layer, (56, 512);
- First bidirectional-LSTM layer, (256, 256);
- Second bidirectional-LSTM layer, (256, 256);
- First fully connected layer, Softmax activation function;
- CTC – loss layer.

As an optimizer, Adam was chosen with a training step of 0.001, the batch size for which there is a one-time adjustment of weights is 64. The length of the alphabet is 43 characters, the maximum possible length of the word is 16 characters. The network weights before training are initialized to normally distributed values with a mathematical expectation of 0 and a standard deviation of 0.1. For extreme pixels in convolutional layers, use automatic zero-fill to get the output layer of the same dimension as the input. The input image was reduced to the size of 31x100 pixels in grayscale, and then fed to the neural network input along with the corresponding word.

3 Experimental and results

Experimental studies were performed on the test set of MNIST and for scene text images separately. For experimental research, a computer program was developed in the Python programming language using the Tensorflow [18] framework.

3.1 Handwritten digits recognition

For experimental studies, the MNIST dataset [19] (Modified National Institute of Standards and Technology) – a database of samples of handwriting numbers was used. Images from this dataset are processed images from another dataset - NIST. NIST samples had a dimension of 20x20 and were taken from the U.S. census Bureau with the addition of test samples written by students of American universities, and then were normalized, smoothed and reduced to the size of 28x28 pixels. MNIST database contains 60000 training and 10000 test gray-scale images of size 28x28 Examples of images are shown in Figure 4.

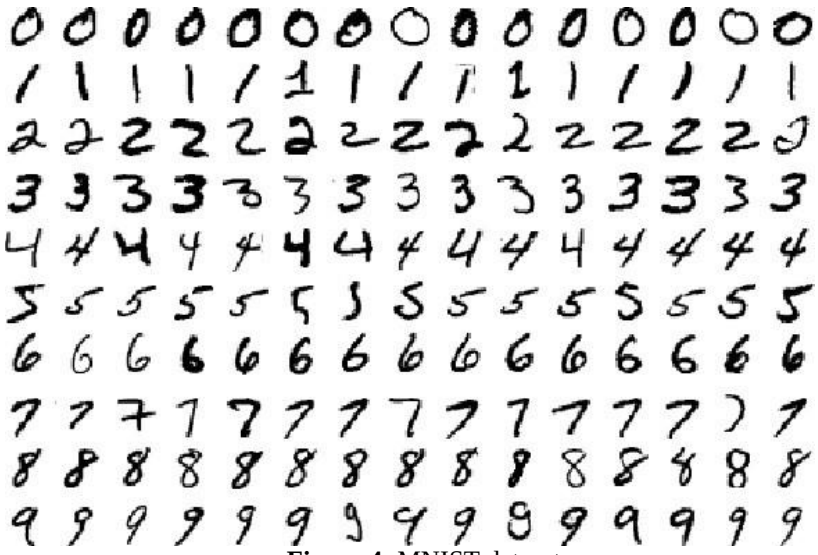


Figure 4. MNIST dataset.

The neural network was trained 20 epochs on the graphics processor NVIDIA GeForce GTX 650, the final accuracy was selected by the best epoch. Before training the neural network, augmentation of the training data set was performed. Each image underwent the following augmentation:

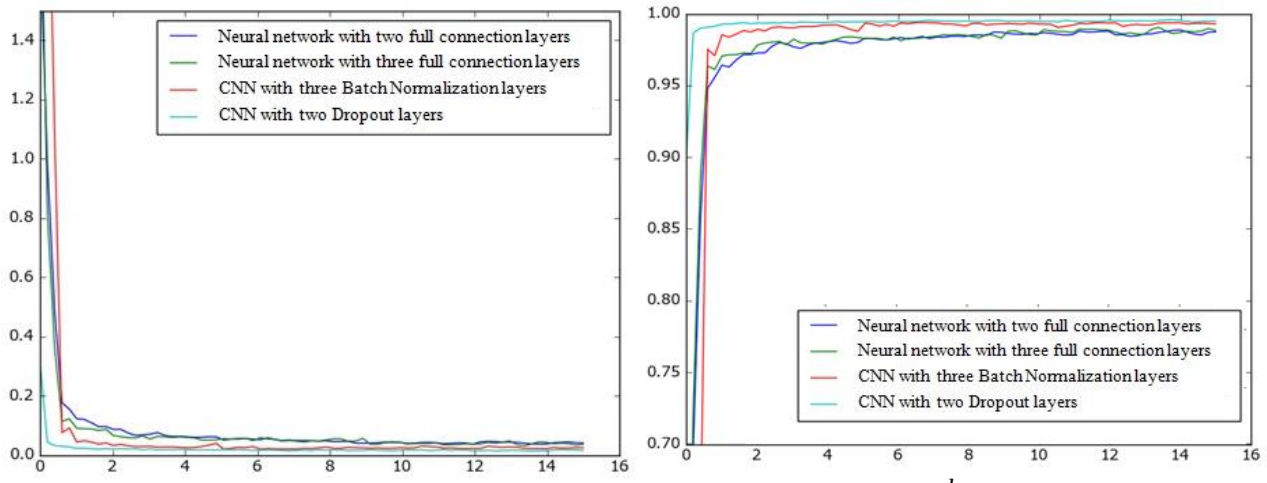
- rotation at a random angle from 0 to 10 degrees;
- random approach and distance in proportions from 0.9 to 1.1 from the original image;
- random horizontal shift in proportions from -0.1 to 0.1 of the width of the original image;
- random vertical shift in proportions from -0.1 to 0.1 of the height of the original image.

Comparative results of experimental studies of the quality of handwritten digit recognition for convolutional neural networks of various architectures are shown in Table 1. The recognition accuracy of handwritten numbers is calculated as the ratio of correctly predicted classes to the total number of sample classes.

Table 1. Handwritten numbers recognition accuracy

Neural networks architectures	Number of optimized parameters	Accuracy, %
Neural network with two full connection layers	407050	98.8
Neural network with three full connection layers	1333770	98.9
Convolutional Neural Network with two Dropout Layers	887530	99.4
Convolutional neural network with three Batch Normalization layers	887978	99.6

Figure 5 shows the results of testing different neural network architectures for the handwriting recognition problem on the MNIST dataset.



a *b*

Figure 5. The results of MNIST experimental studies: a - classification error; b – classification accuracy.

According to the results of the experiment on the test part of the MNIST data set, the maximum accuracy for the best epoch was 99.6%. Such precision showed a convolutional neural network with thee Batch Normalization layers on mini-batchas, this network converges significantly faster than others, while having the highest total precision

3.2 Scene text recognition

In this paper, the dataset Synth 90k, proposed and developed in the article, is used to train neural networks [1]. This dataset is a program-generated image in grayscale, variable width and 31 pixel height. The training part of the set contains more than 7 million images with 90 thousand unique English words. Image Samples from Synth 90k dataset are presented in Figure 6.

Figure 6. Image Samples from Synth 90k dataset.

Two data sets were used to recognize text using a trained network: IIIT 5K [20] and Street View Text - SVT [21]. The IIIT 5K dataset is localized text in photographs of storefronts, billboards, advertisements, street signs, etc. The dataset contains 2000 training and 3000 test images. Image Samples from IIIT 5K dataset are presented in Figure 7.

Figure 7. Image Samples from IIIT 5K dataset.

Images with text on storefronts, billboards, signs of various institutions also prevail in the SVT dataset. This dataset contains 249 images of graphic scenes and 647 images of localized text. Image Samples from SVT dataset are presented in Figure 8.

Figure 8. Image Samples from SVT dataset.

Training sample was 80%, test 20% of the total number of images. The neural network was trained 1 epoch using 7 224 612 images from the Synth 90k dataset. Used graphics processor NVIDIA GeForce GTX 650, the learning process took 110 hours. Figure 9 shows the results of an experimental study to solve the scene text recognition problem using various data sets. The Synth 90k dataset used for training.

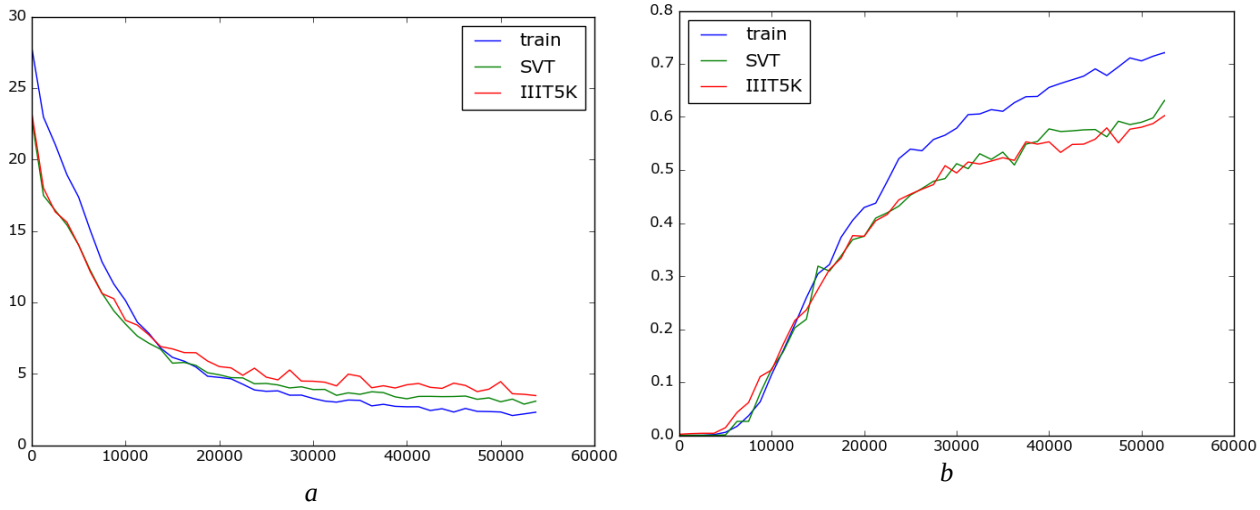


Figure 9. The results of scene text recognition experimental studies: a - classification error; b – classification accuracy.

The comparison of the obtained results scene text recognition accuracy with the results of other authors was implemented using IIIT5K and SVT datasets. The comparative values are placed in Table 2.

Table 2. Comparative results

Method	Dataset				
	IIIT5K			SVT	
	50 words	1000 words	without a dictionary	50 words	without a dictionary
ABBYY [22]	24.3	-	-	35.0	-
Wang et al [22]	-	-	-	57.0	-
Mishra et al. [23]	64.1	57.5	-	73.2	-
Wang et al. [24]	-	-	-	70.0	-
Bissacco et al. [4]	-	-	-	-	78.0
Jaderberg et al. [25]	-	-	-	86.1	-
Jaderberg et al. [1]	95.5	89.6	-	93.2	71.7
Shi et al. (CRNN) [3]	97.6	94.4	78.2	96.4	80.8
FACLSTM [26]	99.5	98.6	90.5	-	82.2
The proposed method	96.7	93.2	74.0	95.3	76.2

The experiments confirm the efficiency of the proposed method for scene text recognitions using the designed CRNN.

4 Conclusions

The paper presents the application of artificial neural network technologies for the problem of text recognition. Comparison of various neural network architectures to recognition for handwritten digits from the MNIST dataset has been carried out. The best accuracy was shown by the convolutional neural network with three Batch Normalization

layers. A further development of this approach is scene text recognition. Scene text images are characterized by the absence of clear criteria for distinguishing the background from the text, the heterogeneity of the background, the high probability of various distortions and noise. Such text can be of variable quality, different font, slope, shape, thickness and texture. The neural network architecture combining convolutional, recurrent layers and CTC error function is proposed for recognition of such text. At the stage of experimental research, the effectiveness of the developed CRNN was compared with the data of other authors. Experimental studies conducted on specialized datasets confirm the validity of the use of technologies convolution neural networks for text recognition tasks.

References

[1] Jaderberg M., Simonyan K., Vedaldi A., Zisserman A. Reading Text in the Wild with Convolutional Neural Networks // International Journal of Computer Vision. 2016. P. 1-20.

[2] Graves A., Fernandez S., Gomez F., Schmidhuber J. Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks // ICML '06 Proceedings of the 23rd international conference on Machine learning. 2006. P. 369-376.

[3] Shi B., Bai X., Yao C. An End-to-End Trainable Neural Network for Image-based Sequence Recognition and Its Application to Scene Text Recognition // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2015. P. 99.

[4] Bissacco A., Cummins M., Netzer Y., Neven H. Photoocr: Reading text in uncontrolled conditions // ICCV. 2013. P. 785-792.

[5] Luo C., Jin L., Sun Z. MORAN: A Multi-Object Rectified Attention Network for scene text recognition // Pattern Recognition. 2019. Vol. 90. P. 109-118.

[6] Zheng Y., Iwana B.K., Uchida S. Mining the displacement of max-pooling for text recognition // Pattern Recognition. 2019. Vol. 93. P. 558-596.

[7] Banerjee I., Ling Y., Chen M.C., Hasa S.A., Langlotz C.P., Moradzadeh N., Chapman B., Amrhein T., Mong D., Rubin D.L., Farri O., Lungren M.P. Comparative effectiveness of convolutional neural network (CNN) and recurrent neural network (RNN) architectures for radiology text report classification // Artificial Intelligence in Medicine. 2019. Vol. 97. P. 79-88.

[8] Gu J., Wang Z., Kuen J., Ma L., Sharoudy A., Shuai B., Liu t., Wang X., Wang G., Cai J., Chen T. Recent Advances in Convolutional Neural Networks // Pattern Recognition. 2018. Vol. 77. Issue C. P. 354-377.

[9] Parkhi O.M., Vedaldi A., Zisserman A. Deep face recognition // Proceedings of the British Machine Vision Conference (BMVC). 2015. Vol. 1. P. 41.1-41.12.

[10] Wang H., He Z., Huang Y., Chen D., Zhou Z. Bodhisattva head images modeling style recognition of Dazu Rock Carvings based on deep convolutional network // Journal of Cultural Heritage. 2017. Vol. 27. P. 60-71.

[11] Kinghorn P. , Zhang L., Shao L. A Hierarchical and Regional Deep Learning Architecture for Image Description Generation // Pattern Recognition Letters // Pattern Recognition Letters. 2019. Vol. 119. P. 77-85.

[12] Alevar-Sandoval R.F., Sanco-Gomer J.I., Figueiras-Vidal A.R. On improving CNNs performance: The case of MNIST // Information Fusion. 2019. Vol. 52. P. 106-109.

[13] Delahunt C.B., Kutz J.N. Putting a bug in ML: The moth olfactory network learns to read MNIST // Neural Networks. 2019. Vol. 188. P. 54-64.

[14] LeCun Y. Learning processes in an asymmetric threshold network. Disordered Systems and Biological Organization. NATO ASI Series (Series F: Computer and Systems Sciences). Springer, Berlin, Heidelberg. 1986. Vol 20.

[15] LeCun Y. Theoretical framework for back-propagation // In Proceedings of the 1988 Connec-tionist Models Summer School. Morgan Kaufmann, CMU. 1988. P. 21-28.

[16] Graves A., Fernandez S., Gomez F., Schmidhuber J. Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks // ICML '06 Proceedings of the 23rd international conference on Machine learning. 2006. P. 369-376.

[17] Hochreiter S., Schmidhuber J.. Long short-term memory // Neural Computation. 1997. P. 1735-1780.

[18] An end-to-end open source machine learning platform. Available at: https://www.tensorflow.org/.

[19] MNIST Database of handwritten digits. Available at: http://www.gavo.t.u-tokyo.ac.jp/~qiao/database.html.

[20] IIIT 5K-word dataset. Available at: http://cvit.iiit.ac.in/research/projects/cvit-projects/the-iiit-5k-word-dataset.

[21] Street View Text (SVT) dataset. Available at: http://vision.ucsd.edu/~kai/svt/.

[22] Wang, K., Babenko, B., Belongie, S. End-to-end scene text recognition // Proc. Int. Conf. on Comp. Vision. 2011. P. 1457-1464.

[23] Mishra, A., Alahari, K., Jawahar, C. Scene text recognition using higher order language priors // Proc. British Machine Vision Conference. 2012, P. 1-11.

[24] Wang, T., Wu, D.J., Coates, A., Ng, A.Y.: End-to-end text recognition with convolutional neural networks // ICPR. 2012 P. 3304-3308.

[25] Jaderberg, M., Vedaldi, A., Zisserman, A. Deep features for text spotting // European Conference on Computer Vision. 2014. P. 512-528.

[26] Wang Q. Jia W., He X., Lu Y., Blumenstein M., Huang Y. FACLSTM: ConvLSTM with Focused Attention for Scene Text Recognition // Computer Vision and Pattern Recognition. 2019. P. 4321-4329.