Journal of Theoretical and Applied Information Technology
31st July 2020. Vol.98. No 14

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

KAZAKH HANDWRITTEN RECOGNITION

[1]BEIBUT AMIRGALIYEV, [2,3]ARMAN YELEUSSINOV, [4]MIYACHI TAIZO,

[1]Astana IT University, Kazakhstan

[2]Institute of Information and Computational Technologies CS MES RK, Kazakhstan,

[3]Al-Farabi Kazakh National University, Kazakhstan

[4]Tokai University, IT Education Center, Japan,

E-mail: [1]beibut.amirgaliyev@astanait.edu.kz,[2]armankaznu@mail.ru, [3]miyachi@tsc.u-tokai.ac.jp

ABSTRACT

Recognition of handwritten texts in different languages has recently attracted great interest among researchers in connection with the development of various algorithms for machine learning, deep learning and computer vision. This paper proposes a labeling approach for collecting dataset for handwritten Kazakh text recognition (HKTR). We consider the stages of creating a handwritten database for the Kazakh language and preparing a training sample for training a neural network. The results of applying widely used machine learning algorithms for recognizing offline handwritten texts are presented.

We evaluate the performance of the proposed method on the test sets from our Kazakh handwritten dataset and Russian handwritten dataset, and achieve performance with correct rate 85.63.

Keywords: *handwriting recognition, deep learning, SVM, CNN.*

1. INTRODUCTION

Handwriting recognition is one of the most popular and active research in the field of pattern recognition. Autonomous handwriting recognition systems are very important for creating electronic libraries, mail sorting, digitalizing handwritten documents and historical books. Document text recognition facilitates automatic data entry applications by identifying the content of text in document images and thus significantly reduces manual effort. Progress in this area is due to the achievements in the development of many works and the availability of an international database that allowed researchers to test the effectiveness of their approaches and compare them with others using the same databases. The first step in implementing manuscript recognition for any particular language is to select a suitable database of text images. If such databases are not available, then the first task is to create a new database of manuscript images for this language. Researchers in the field of text recognition have developed several databases for evaluating OCR systems for offline handwriting recognition. Several of these databases are available for texts in Latin, French, Chinese, Korean, Japanese, Arabic, Uyghur, Urdu, Hindi, Spanish, Tamil, Indonesian, Greek, Russian. These databases differ from each other in their structure and content. Some consist of letters, some of which consist of words or phrases [1-10]. As follows from the database review for recognizing the base of handwritten characters, there are no words in the Kazakh language.

The task of text recognition has two stages: HTR and OCR. Handwriting recognition (HTR) task with words in handwriting without emphasis on individual letters, and optical character recognition (OCR) recognizes each letter independently. In this article, we will look at the HTR task.

Machine learning, although the name has existed since the 1950s, only recently has this area been developed and used in various fields of activity. New and wider applications of these methods in everyday life also appear due to the use of properties that were previously unavailable due to their complex and innovative structure - image recognition, classification, prediction of human behavior and decision-making, geology, Oil & Gus, Control System, Robotics, etc. [11-13]. This field has a bright future because it can facilitate decision making or even replace people in some areas.

Recognition of handwritten characters of the Kazakh language remains a difficult problem due to the specifics of the Kazakh language. The Kazakh alphabet contains 42 letters with uppercase and

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

lowercase letters. The shape of each character in italic text changes depending on the adjacent characters. There is also a huge variety of writing styles for individual authors [Figure 1]. It is a proven fact that different handwriting has a decisive influence on performance and recognition speed. Therefore, there is a need to create databases of Kazakh characters of different authors for training, testing and comparison of recognition systems.

After detecting text in an image, the next important task is text recognition. Recognition systems imitate human reading, processing the image of a text string as a sequence of frames. Elements are extracted from each frame and transferred to a system that converts them into a character string. Despite extensive research with hidden Markov models (HMMs) and hybrid neural network-HMMs for sequential transcription of data [1] - [3], feature extraction remains a problem. The traditional stand-alone HCR method typically uses an architecture of three main phases: preprocessing the entered handwritten character, extracting features, and classifying using machine learning techniques such as Support Vector Machine (SVM), Multi-Layer Perceptron (MLP), Random Forest (RF), decision trees, etc.

This paper is an extended work of conference proceeding IV international scientific and practical conference "Computer science and applied mathematics", 2019 [25]. The lack of a standard reference database of handwritten symbol images for the Kazakh language makes it difficult to compare the research results. In this work, we present the stages of creating a database of images of symbols of the Kazakh language and analysis of the effectiveness of various recognition systems in the created database.

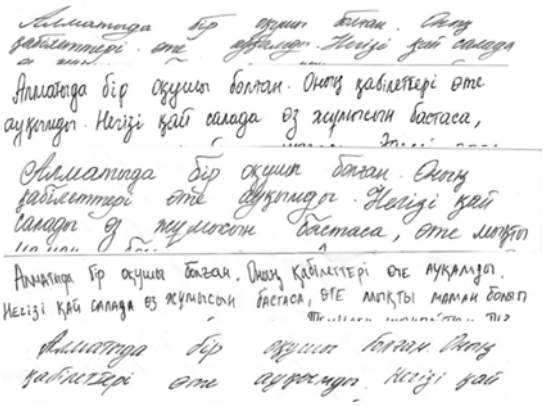

Figure 1. Various samples of Kazakh handwritten texts.

2. DATA COLLECTION AND PROCESSING

This section describes the process of collecting handwritten image data to create a database of Kazakh characters. The reasons why HKCR is a nontrivial problem can be formulated as follows:

- spelling variation;

- italic text;

- the similarity between the characters.

The main task in the implementation of the recognition system is the collection of data samples. Data collection is one of the tedious tasks in most pattern recognition applications. Character classes are defined based on the spelling structures of the Kazakh language. The Kazakh alphabet based on the Cyrillic alphabet consists of 42 lowercase and uppercase letters, of which 33 are letters of the Russian alphabet and 9 specific letters of the Kazakh language. Each letter has different forms, depending on its position in the word, which can be at the beginning, in the middle or at the end. Therefore, to achieve the accuracy of the system, the characters are not written separately, but are extracted from the written words. Handwritten texts are collected from various 120 local authors. To collect images of characters, authors are invited to write texts from various sources on the pages. Collected handwritten data sheets are scanned and saved in png format. In Figure 2 shows handwritten samples and marked character classes.

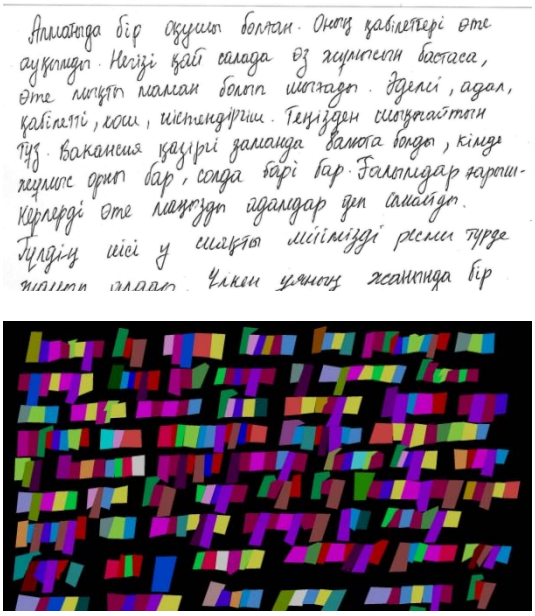

Figure 2. a) an example of handwritten text b) labeled polygons.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Segmented images of characters are manually grouped into the considered 80 classes of the Kazakh language. Heavily distorted images are excluded during grouping, and segmented symbolic images are resized to 32x32. Information about them is written to the csv file. Each line in csv files represents a character image. The first column represents the character, the second represents the class label, and the third column represents the image number. For the training data set, 75% of the symbol images from each class were taken, the remaining 25% make up the test data set. The total number of images in the created database is 59,274. The number of images of characters in the class is not the same, as some characters are rare, and some images are discarded due to incorrect character segmentation. The average number of symbolic images for a class across the entire database is 1340. Below Figure 3 shows samples taken from 42 classes of handwritten characters of the Kazakh language.

Figure 3. Randomly selected character images in 42 different classes.

3. PRE-PROCESSING AND CLASSIFICATION USING MACHINE LEARNING

3.1 Data preprocessing

Traditional offline OCR methods typically use architecture of three main steps: pre-processing the entered handwritten character, extracting features, and classifying using machine learning methods such as support vector methods (SVM), multi-layer perceptron (MLP), k-nearest neighbors (KNN), random forest (RF), decision trees, etc.

To obtain the best result and remove noise, the stage of preliminary processing of symbol images in the database is performed. When recognizing a manuscript using feature identification algorithms that extract feature vectors from complex images of handwritten characters, namely, scale-invariant character transformation (siftD) and directional gradient histogram (HOG), relatively good results were obtained in different datasets [14]. Object descriptors compute object vectors using image filters, such as the Sobel and Gaussian filters. Subsequently, the orientations within each area are calculated and weighted in the histogram direction. Since these feature descriptors are invariant to small local offsets, the descriptors provide reliable feature vectors.

3.2 Histogram of oriented gradients

Typically, histogram of oriented gradient (HOG) is used as representing the exact location of an image for detecting an object in various areas, such as the detection of persons, pedestrians, and vehicles [15]. The HOG descriptor is initially defined as the distribution of local intensity gradients from the image, which are calculated from small related areas (cells). To increase the efficiency of searching for objects, it is necessary to normalize the colors and gamma of the image.

The HOG feature vector is computed from the image using gradient detectors. After dividing the source images into blocks, each block is divided into small areas known as cells. Each pixel is collapsed with a simple convolution kernel as follows:

$$\begin{aligned} G_x &= f(x+1,y) - f(x-1,y) \\ G_y &= f(x,y+1) - f(x,y-1) \end{aligned} \quad (1)$$

G_x and G_y are the horizontal and vertical component gradients, respectively. In our experiments, the HOG descriptor is calculated over rectangular blocks (R-HOG) with non-overlapping blocks. The magnitude of the gradient M and the orientation of the gradient θ are calculated as:

$$\begin{aligned} M(x,y) &= \sqrt{G_x^2 + G_y^2} \\ \theta(x,y) &= \arctan\left(\frac{G_y}{G_x}\right) \end{aligned} \quad (2)$$

Then, for each cell, a HOG is generated that spans the entire image. In the end, function descriptors are normalized by applying the normalization of L2 blocks as follows [16]:

$$V'_k = \frac{V_k}{\sqrt{\| V_k \|^2 + \grave{o}}} \quad (3)$$

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

where V_k is the combined histogram from all areas of the block, $\grave{o}$ is a small value close to zero, and V'_k is the normalized feature vector of the HOG descriptor.

In our case, the gradients are calculated in the range of [0.180]. Histograms of 8 bins are calculated with values in the form of weights. For calculation, each image is normalized to a size of 32x32 and by gamma, and then by contrast (Figure 4).

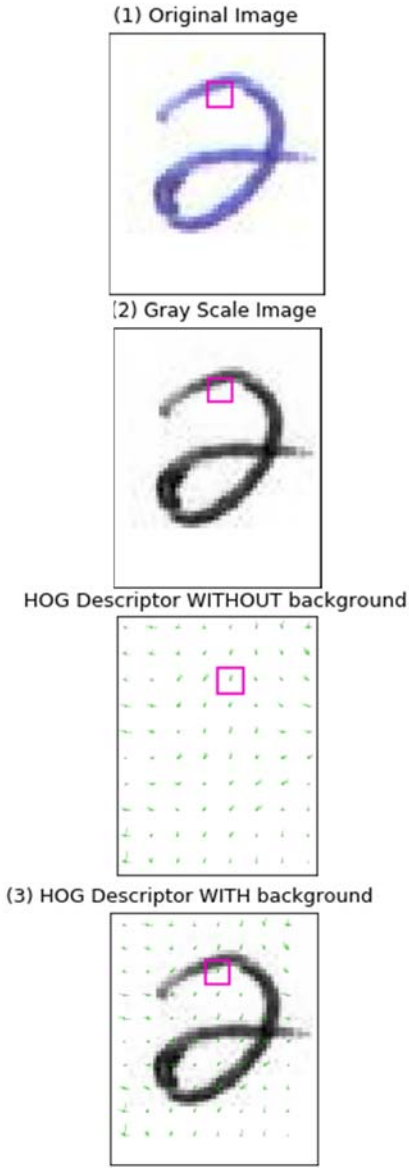

Figure 4. Visualization of the result of the HOG descriptor.

Each pixel matrix of the 32x32 image is divided into 8x8 cells, and then a 4x4 matrix with 8 cells in each cell is obtained. This matrix was obtained as 2X2 blocks (with an overlap of 50%) and normalized by dividing the histogram bin vector by the value. A total of 9 blocks x 4 cells x 8 bins = 288 signs.

Computation of the HOG descriptor requires the following basic configuration parameters:

- ✓ Masks to compute derivatives and gradients.
- ✓ Geometry of splitting an image into cells and grouping cells into a block.
- ✓ Block overlapping.
- ✓ Normalization parameters.

The result of the histogram of the gradients is shown in Figure 5.

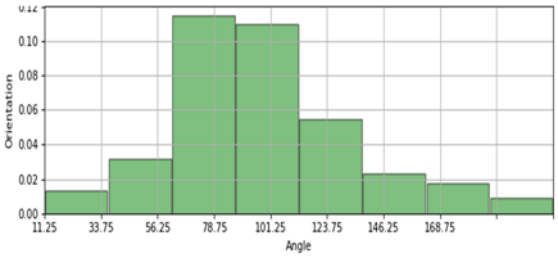

Figure 5. Histogram of gradients.

3.3 Support vector machine

The support vector method (SVM) is effectively used to solve many pattern recognition problems [17-18]. This algorithm outputs the optimal hyper plane and maximizes the difference between the two classes. To solve the problem of nonlinear data in SVM there is a nuclear method that allows you to get higher accuracy.

The linear SVM algorithm has been expanded to address nonlinear classification problems. Many nonlinear kernel functions have been proposed. In this article, we select the core of the radial basis function (RBF) as a nonlinear function in the SVM classifier. This is also called the Gaussian core. The RBF kernel computes the following value between two input vectors:

$$K(x, x') = exp\left(\frac{-\|x - x'\|^2}{2\sigma^2}\right) \quad (4)$$

where $1/2\sigma^2$ is the RBF core parameter. Large parameter values can cause over fitting due to an increase in the number of reference vectors.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

For training the SVM classifier, we will provide all the necessary data (labels, training and testing data) from above as input. This usually takes the most computational time. Based on a review of the provided tagged data, SVM classifies unknown images. In order to obtain the accuracy of a trained classifier, the characters identified by SVM correspond to the provided training marks.

The influence of the parameters of the RBF function on the results was considered in [19-20]. Gamma parameters can be considered as the reciprocal of the radius of influence of the samples selected by the model as reference vectors. Parameter C replaces the correct classification of training examples and maximizes the margin of the decisive function. Teaching the results of the above works and that searching the grid is a very time-consuming process, in this article we use parameters in the range C = [1.5], gamma = [0.01.0.05].

4. DEEP LEARNING CLASSIFICATION

With the development of deep learning in recent years, convolutional neural networks (CNNs) have led to new breakthrough technologies for manuscript recognition with great success [21, 22]. CNNs utilize many multilayer perceptions that are supposed to have a slight predisposition. Also called artificial neural networks, shear invariant or domain invariants, in accordance with their architecture with general weights and transformation dispersion properties. Modern CNNs used in machine learning are built using the same principles as traditional ones: they use alternating convolution (CL) layers and maximum pool (MP) layers, followed by a series of fully connected (FC) layers. The most common CNN model architectures are: LeNet, AlexNet, ZF Net, GoogLeNet, VGGNet, and ResNet.

4.1 VGG model

One of the most popular deep convolutional architectures is a model called VGG. The name comes from the fact that this model was developed at the University of Oxford in the Visual Geometry Group. VGG is actually two configurations of convolutional networks at once, on 16 and 19 layers. The main innovation compared to previous architectures was the idea of using filters of size 3-3 with a single convolution step instead of convolutions with filters 7-7 with steps 2 and 11-11 with step 4 used in the best models of previous years. The resulting model in 2014 won one of the nominations for the renowned computer vision competition, ImageNet Large Scale Visual Recognition Competition (ILSVRC).

4.2 GoogleNet model

The next convolutional architecture is the Inception architecture. It was developed by Google and appeared almost simultaneously with VGG, in September 2014. The basic idea is to use not just a convolution – nonlinearity – subsampling sequence as building blocks for deep convolutional networks, as is usually done, but more complex constructions. In Inception, such a component is “assembled” from small convolutional structures. So the name of the network reflects the idea of “embedded” architecture developed there. This work contains several extremely interesting and important ideas, due to which, in particular, despite the large declared depth - 22 layers without taking into account subsampling - Inception actually has fewer parameters than VGG.

As practice has shown, it has become possible to teach deep architectures efficiently, however, those solutions to which neural networks of great depth converged often turned out to be worse than for less deep models. It turned out that this is a more fundamental problem: with the addition of new layers, the error grows not only on the test, but also on the training set.

4.3 ResNet model

To solve the problem of degradation, a team from Microsoft Research developed a new idea: deep residual learning, which formed the basis of the ResNet network, as well as many subsequent works. There is nothing new in the basic structure of the new model: these are layers that follow one after another. Separate levels, building blocks of the network also look quite standard, these are just convolutional layers, usually with additional normalization to mini-batches. The difference is that in the residual block the layer of neurons can be “circumvented": there is a special connection between the output of the previous layer $x(k)$ and the next layer $x(k+1)$, which goes directly, not through a layer computing something.

All this makes it possible to train very, very deep networks. Kaiming He calls this the “revolution of depth”: in VGG there were 19 levels, in GoogLeNet - 22 levels, in the first version of ResNet - immediately 152, and in the latest versions of networks with residual connections, networks can be trained up to a thousand levels in depth without problems.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

4.4 Our model

In our architecture, the input image size is 32 × 32. A tensor with sizes of 1024 pixels each is used as input. We started by loading the data and transforming it into an array of images with 4 dimensions. Our neural network consists of 17 layers. The first convolutional layer has 5 × 5 cores, and the remaining convolutional layers 3 × 3. The second convolutional layer converts the 32-dimensional feature vector to 64, and the third to 128. We use 2 × 2 pooling layers. Pooling layers use a non-overlapping 2x2 join, so down sampling occurs here. Then, in a fully connected to each input, each node in the next layer will be connected, instead, some random edges should be discarded. This is done in the dropdown layer, and this is important because congestion should be avoided. All hidden layers use the ReLU (Rectified Linear Unit) activation function.

First of all, we note that ReLU neurons are more efficient than those based on logistic sigmoid and hyperbolic tangent. The following ideas that have begun to gain popularity recently are various modifications and generalizations of ReLU that try to maintain computational efficiency, but at the same time add a little flexibility to the basic design. This is LeakyReLU. The idea here is to try to improve optimization, because if all the gradients are strictly zero on the negative part of the definition domain, these neurons will not be trained at all. It is shown that in those constructed on LeakyReLU's deep acoustic speaker networks improve speech recognition. For this reason, we use it in the proposed architecture of neural network.

For measurement effectiveness of the neural network we cchoose a metrics. The result of correct recognition of handwriting is binary decision (yes/no), we choose Precision and Recall metrics. And they are calculated base on values such the number of results True positive, True negative, False positive, False negative.

$$\mathbf{Precision} = \frac{\mathbf{True\ positives}}{\mathbf{True\ positive + False\ positives}},$$

$$\mathbf{Recall} = \frac{\mathbf{True\ positives}}{\mathbf{True\ positive + False\ negatives}}$$

The proposed architecture and complete information is presented in table-1. The final convolution block creates an 8 × 8 × 128 object map. The output layer will have 42 nodes, each node will take a value from 0 to 1, and it will represent each character. Then it is fed to the output of 42-softmax, where the number corresponds to the number of character classes considered in this work.

Table 1: The architecture used for classify handwritten characters.

Layer	*Dimension*	*Parameters*
conv2d_1(Conv2D)	32, 32, 32	832
max_pooling2d_1 (MaxPooling2)	16, 16, 32	0
dropout_1 (Dropout)	16, 16, 32	0
conv2d_2 (Conv2D)	16, 16, 64	18496
leaky_re_lu_1 (LeakyReLU)	16, 16, 64	0
dropout_2 (Dropout)	8, 8, 64	0
conv2d_3 (Conv2D)	8, 8, 128	73856
leaky_re_lu_2 (LeakyReLU)	8, 8, 128	0
max_pooling2d_3 (MaxPooling2)	4, 4, 128	0
dropout_3 (Dropout)	4, 4, 128	0
global_max_pooling2d_1 (Glob)	128	0
dense_1 (Dense)	1024	132096
leaky_re_lu_3 (LeakyReLU)	1024	0
dropout_4 (Dropout)	1024	0
dense_2 (Dense)	42	13325
activation_1 (Activation)	42	0

During training the neural network we applied nest parameters:

- learning rate=0.001
- epochs=100
- batch_size=2359
- loss function: binary_crossentropy

For binary classification, the function that we will minimize on the dataset $D = \{(x_i, y_i)\}_{(i=1)}^{N}$, in general looks like average cross entropy function for the dataset:

$$L(\theta) = -\frac{1}{N}\sum_{i=1}^{N}\left(y_i log\hat{y}_i(\theta) + (1-y_i) log\left(1-\hat{y}_i(\theta)\right)\right) \quad (5)$$

where $\hat{y}_i$ - answer from the neural network.

As a result, we obtain a continuous function $L(\theta)$ from the probabilities predicted by the classifier, which estimates how well it predicts labels in the data. $L(\theta)$ is as a function of error; it can be differentiated and optimization will work.

5. EXPERIMENTS AND RESULTS

This section briefly discusses some comparisons between our experimental CNN and

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

HOG feature based SVM architectures. Also, after training the model, we will compare the results on two datasets, which are collected by different approaches.

Modeling and experiments were carried out on the Jupyter Notebook platform. Character recognition efficiency is evaluated for our dataset of handwritten characters of the Kazakh language using various algorithms for distinguishing and classifying characters.

5.1 Classification with SVM

As in the normal image recognition process, the dataset is divided into training and test sets. In our experiments, we used 42 classes of sets that have normal images for determining performance in algorithms. Figure 6 shows the number of training and test samples for each class. In our experiment, we used 80% of the data for training, and the other 20% for testing. The validation performance was measured by splitting the initial training set into 70% for training and the remaining 30% for validation. The recognition rate was evaluated using the following formula:

$$\text{Recognition rate} = \frac{Correct}{Total}$$

The training was stopped when there was no improvement in the error rate of validation set for 40 consecutive epochs. The classification rates read at 92.40% and 85.63% on training and validation sets respectively on epoch 135. Table 2 summarizes the character error rates on training set and validation set.

Table 2: Performance metrics SVM

	Training dataset	Validation dataset
Precision, %	92.40	85.63
Recall, %	81.14	78.66

HOG features are computed by dividing the image window into small regions called “cells”, and a local one-dimensional histogram of gradient directions is computed for each cell. The features so obtained have shown certain degree of tolerance to geometric transformations.

5.2 Classification with CNN

The feature extraction process extracts informative descriptors from images and helps classifiers in assessing decision boundaries among participating classes. To protect the proposed model from retraining, we used the exclusion or dropout method as a network optimization method [23].

In the training process, we noticed that after 70th epoch, the model began to show signs of retraining, so the end of training occurred at this epoch the loss of our network was 0.6, and the overall accuracy of the verification data was 0.82. Figure 7 shows how the training loss and validation of our model changed during the training process. Figure 8 shows how the accuracy of the model has changed in the training and validation data. Table 3 summarizes the character error rates on training set and validation set with CNN.

Table 3: Performance metrics CNN

	Training dataset	Validation dataset
Precision, %	95.44	86.47
Recall, %	85.21	80.65

5.3 Comparison results

Since the Kazakh alphabet is based on the Russian alphabet, in our experiment we decided to use the dataset of the Russian handwritten dataset for comparison. Because its formation is different from our approach. The basis for the formation of Russian dataset are forms filled out by means of a survey of respondents, which are subsequently scanned and processed and segmented to highlight isolated images [24]. Therefore, the process of forming a dataset can be divided into the following stages:

- designing the structure of forms and interviewing respondents.
- processing completed forms.
- developing segmentation procedures and a user interface for accessing data.

Figure 9 below shows example of a completed form.

Journal of Theoretical and Applied Information Technology
31st July 2020. Vol.98. No 14

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

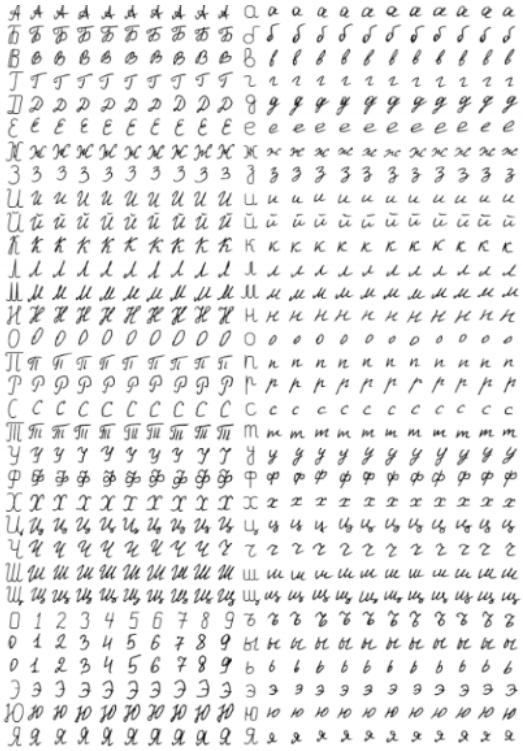

Figure 9. Example of a completed form

In the process of segmentation used an advanced method of segmentation based on histograms with an algorithm for subsequent localization of the position of the symbol within the marking cell. The procedure for segmenting an image into individual characters consists of the following basic operations:

- building a binary image of a grid of vertical and horizontal histograms.
- segmentation of a binary image of characters:
  a) threshold binarization of the constructed histograms.
  b) the determination of the coordinates of the borders of cells from the found maximum points of the binarized histograms;
- character extraction.
- marking and saving the extracted characters.

For word recognition, we use the trained models described in the previous sections. In order to investigate the potential of using training for offline cursive handwriting recognition we compared several handwritten texts. Below is an example of handwriting and segmentation result.

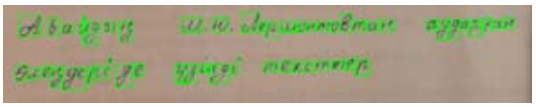

Figure 10. Example of segmented text

As a result of recognition in the context of a word, only 70% characters are correctly identified, while isolated letters are recognized with about 90%. The table 4 below shows examples of the results of processing the word "Lermontov". The highlights those characters that are recognized correctly. Most often in this word the letters "e" and "o" were correctly identified.

Table 4: Performance metrics CNN

Model/dataset	Л	е	р	м	о	н	т	о	в
SVM/Kazakh	**л**	**е**	**р**	ш	**о**	**н**	м	**о**	б
CNN/Kazakh	**л**	**е**	**р**	**м**	**о**	**н**	**т**	**о**	в
SVM/Russian	п	**е**	п	ш	а	**н**	**т**	**о**	г
CNN/Russian	**л**	**е**	**р**	ш	**о**	**н**	**т**	**о**	б

6. CONCLUSION

In this paper, we presented a new approach of the collecting of the dataset. New image dataset of the Kazakh handwritten letters was prepared and tested by SVM and CNNs for the letter recognition. The dataset was published for the data science community as an open source resource. of handwritten characters of the Kazakh language. There was a comparative experiment of offline character recognition algorithms. According to the results of experiments, it was found that the efficiency level of the CNN was much higher than that of the HOG feature based SVM algorithm. Experimental results demonstrate the efficacy of our method, showing usual performance.

In the experiment, our system performed much better on our dataset than Russian because of the unbalanced per-character sample distribution on the datasets. In future works, we plan to expand our database with background images for investigating other CNN architectures such as ResNet and AlexNet. We also schedule to perform on the neural ordinary differential (ODE) equation and investigate its properties in HTR task for Kazakh language. Neural ODE has several advantages over CNN (like ResNet) and have prospects to improve speed and save memory during the training process.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

ACKNOWLEDGMENTS

Funding: This work is supported by a grant from the Ministry of Education and Science of the Republic of Kazakhstan within the framework of the Project "AP05132648 - Creating verbal and interactive robots based on advanced voice and mobile technologies"

REFERENCE LIST

[1] J. J. Hull, "A Database for Handwritten Text Recognition Research", *IEEE transactions on pa'itern analysis and machine intelligence*, Vol. 16, no. 5, May 1994, pp. 550-554

[2] E. Grosicki, H. El-Abed, "Icdar 2011-french handwriting recognition competition", *In Document Analysis and Recognition (ICDAR)*, 2011 International Conference on, IEEE, 2011, pp. 1459-1463.

[3] L. Jin, Y. Gao, G. Liu, *et al.* "SCUT-COUCH2009 — a comprehensive online unconstrained Chinese handwriting database and benchmark evaluation", *IJDAR*, Vol. 14, 2011, pp. 53–64.

[4] K. Dae-Hwan, Y. Hwang, P. Sang-Tae, K. Eun-Jung, P. Sang-Hoon, B. Sung-Yang. "Handwritten Korean character image database PE92", *IEICE TRANSACTIONS on Information and Systems*, Vol.E79-D, No.7, 1996, pp.943-950.

[5] M. Nakagawa, X. Zhou, C. Liu, D. Wang, "Handwritten Chinese/Japanese Text Recognition Using Semi-Markov Conditional Random Fields", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 35, no. 10, Oct. 2013, pp. 2413-2426.

[6] A. Mezghani, S. Kanoun, M. Khemakhem, H. E. Abed, "A database for arabic handwritten text image recognition and writer identification", *Frontiers in Handwriting Recognition (ICFHR),* 2012 International Conference on, IEEE, 2012, pp. 399-402.

[7] K. Ubul, M. Zunun, A. Aysa, N. Yadikar, U. Yunus, "Creation of uyghur offline handwritten database", *Systems, Signal Processing and their Applications (WoSSPA)*, 2013 8th International Workshop on, IEEE, 2013, pp. 291-295.

[8] M. Sagheer, C. He, N. Nobile, C. Y. Suen, "A new large urdu database for off-line handwriting recognition", *In Image Analysis and Processing,* ICIAP 2009 , pp. 538-546.

[9] B.V.R.G.Benne, "Kannada, Telugu and Devanagari Handwritten Numeral Recognition with Probabilistic Neural Network : A Novel Approach", IJCA Special Issue on "Recent Trends in Image Processing and Pattern Recognition" , 2010

[10] M. R. Hussain, A. Raza, I. Siddiqi, K. Khurshid & C. Djeddi, "Erratum to: A comprehensive survey of handwritten document benchmarks: structure, usage and evaluation", *EURASIP Journal on Image and Video Processing,* Vol. 2016, No. 38, 2016.

[11] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, "Gradient-based learning applied to document recognition", *Proceedings of the IEEE*, Vol. 86, No. 11, 1998, pp. 2278-2324.

[12] T. Merembayev, R. Yunussov, Y. Amirgaliyev, "Machine Learning Algorithms for Classification Geology Data from Well Logging", *In 2018 14th International Conference on Electronics Computer and Computation (ICECCO)*, IEEE., 2018, pp. 206-212.

[13] Y. Amirgaliyev, S. Shamiluulu, T. Merembayev, & D. Yedilkhan. "Using Machine Learning Algorithm for Diagnosis of Stomach Disorders". *In International Conference on Mathematical Optimization Theory and Operations Research*, Springer, Cham, July 2019, pp. 343-355.

[14] O. Surinta, M. Karaaba, R.B. Schomaker, M. A. Wiering, "Recognition of handwritten characters using local gradient feature descriptors", *Engineering Applications of Artificial Intelligence*, Vol. 5, October 2015, pp. 405–414.

[15] O. Deniz, G. Bueno, J. Salido, F.D. Torre, "Face recognition using histograms of oriented gradients", *Pattern Recognit. Lett.* Vol. 32 No. 12, 2011, pp. 1598–1603.

[16] S.E. Lee, K. Min, T. Suh, "Accelerating histograms of oriented gradients descriptor extraction for pedestrian recognition", *Comput. Electr. Eng.*, Vol. 39, No. 4, 2013, pp. 1043–1048.

[17] L. Botton, C. Cortes, J.S. Denker, "Comparison of Classifier Methods: Case Study in Handwritten Digit Recognition", *Proceedings of the 12th IAPR International Conference on Pattern Recognition*, USA, California, Vol.3, 1994.

[18] H. Zhonga, J. Wang, H. Jiac. Y. M. Shilei, "Vector field-based support vector regression for building energy consumption", Applied Energy, Vol. 242, 2019, pp. 403-414.

[19] Vempati, Sreekanth and Vedaldi, Andrea and Zisserman, Andrew and C. V. Jawahar,

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

"Generalized rbf feature maps for efficient detection", *Proceedings of the British Machine Vision Conference*, BMVA Press, September 2010, pp. 1-11.

[20] A. Vedaldi and A. Zisserman. "Efficient additive kernels via explicit feature maps", in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 34, no. 3, March 2012, pp. 480-492.

[21] Z. Zhong, L. Jin and Z. Xie, "High performance offline handwritten Chinese character recognition using GoogLeNet and directional feature maps," *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*, Tunis, 2015, pp. 846-850.

[22] E. Chammas, C. Mokbel and L. Likforman-Sulem, "Handwriting Recognition of Historical Documents with Few Labeled Data," *2018 13th IAPR International Workshop on Document Analysis Systems (DAS)*, Vienna, 2018, pp. 43-48.

[23] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors", *ArXiv* abs/1207.0580, 2012: n. pag.

[24] N. A. Fedin, A. Yu. Istratov, "Formation of the base of segmented hand-written the Russian language symbols", *Vestnik komp'yuternykh i informatsionnykh tekhnologiy,* 2013. № 7. pp. 36-41.

[25] Y. Amirgaliyev, A. Yeleussinov, Sh. Mazhitov, "About the development of the database of handwritten symbols of the kazakh language", IV ISP conference "Computer science and applied mathematics", 2019, №1, pp. 157-164.

.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

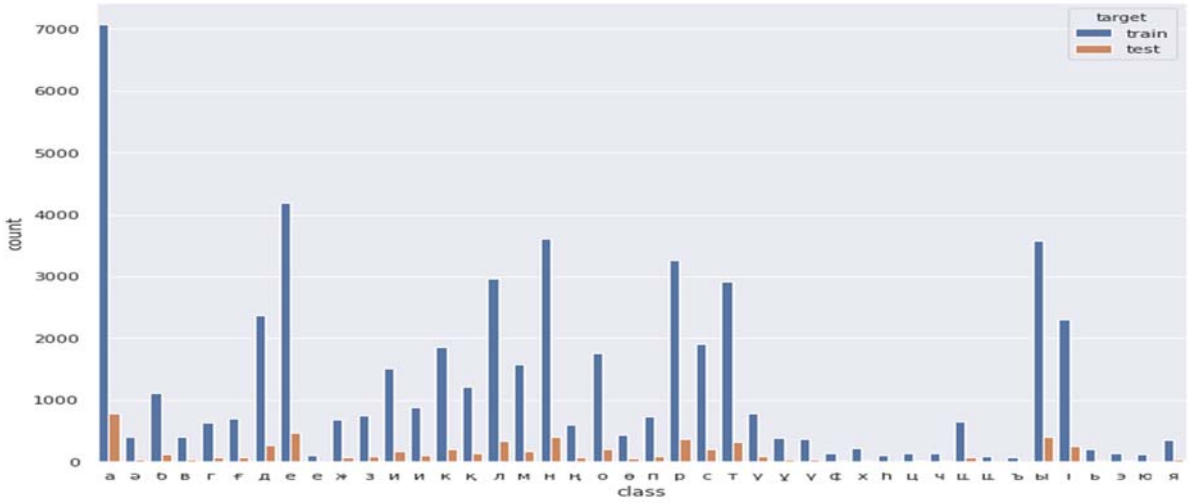

Figure 6. The amount of training and test data for each class

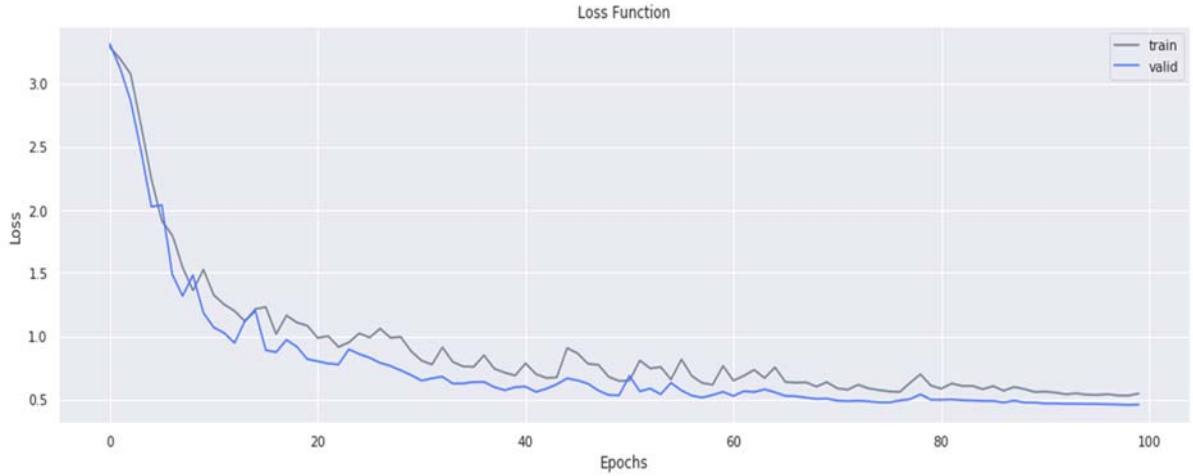

Figure 7. Training and validation loss during training process.

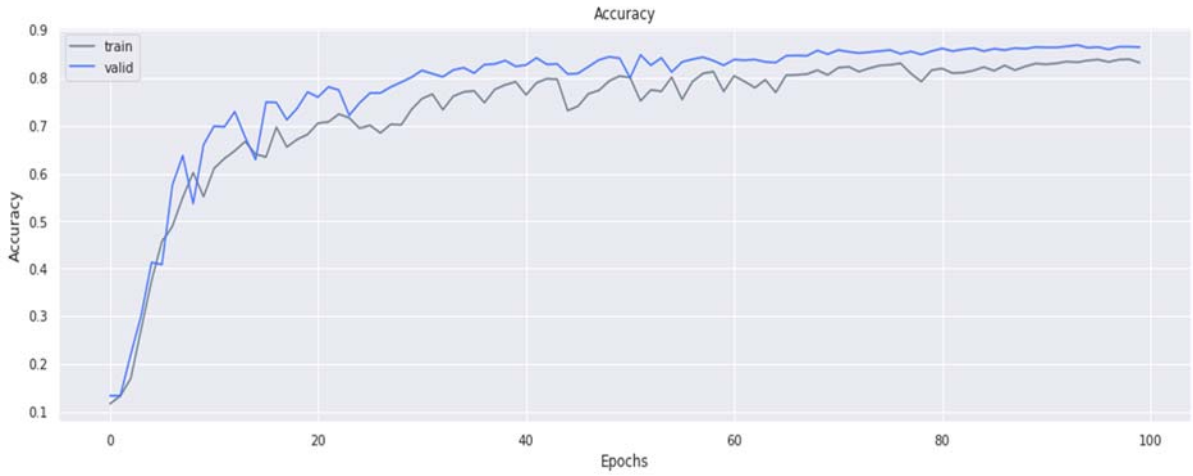

Figure 8. Accuracy of validation and training data.