SATBAYEV UNIVERSITY

Computing & Engineering

Volume 1 (2023), Issue 3, 36-41

https://doi.org/10.51301/ce.2023.i3.07

Recognition of latin letters of the kazakh alphabet based on a neural network

P.M. Pirniyazova*, E.Yu. Son, D.Zh. Kulzhan

Satbayev University, Almaty, Kazakhstan

**Corresponding author: pirniyazova1974@gmail.com*

Abstract. This article discusses the recognition of Latin letters of the Kazakh alphabet based on a neural network using the Tensorflow framework. The paper provides an overview of existing recognition methods, including methods based on machine learning. To do this, a database of Kazakh letters of the Latin alphabet is being created using the Turtle graphic library. A computational algorithm for recognizing Kazakh letters of the Latin alphabet has been developed. The algorithm uses operations on labels entered for letters to predict the target variable. The model is trained on the prepared data and then evaluated based on its performance on the validation dataset. The overall accuracy of the model is approximately 93.90%. This means that about 93.90% of the model's predictions were correct. The paper uses two metrics for the accuracy Precision and completeness Recall classes. These metrics show whether the models work well by class, show information about the performance of the model. The experiments have shown that the proposed algorithm provides high accuracy of recognition of letters of the Latin Kazakh alphabet. The results are illustrated graphically. In the discussion of the results, a multiclass ROC curve is presented, which is a graphical representation of the performance of the classification model at all classification thresholds. The performance of the model as a whole indicates the high performance of the classification model. The algorithm proposed in the article for recognizing Latin letters of the Kazakh alphabet can be used in various applications, such as optical character recognition (OCR) systems, automated verification of entered texts using mobile devices.

Keywords: *recognition, neural networks, perceptron, synaps, weight matrix, pixel matri, layers, errors.*

1. Introduction

The growth of digital technologies opens up many opportunities for working with data. The gradual transition of the Kazakh language to the Latin script is a challenge to modernity and changes in modern realities, that is, there is a place of identity and culture of the language. Looking into the history and culture of the language, its changing history of Romanization began at the beginning of the 20th century of the last century. The idea of the transition of the Latin script of the Kazakh alphabet of that time was to raise the culture of the language and literacy of the population. The gradual transition then began as early as 1923 and officially passed in 1929 of the last centuries. The Latin alphabet of that time consisted of 30 letters with the need to add signs to them to give specificity to the sounds of the Kazakh alphabet. The life of the Latin alphabet at that time was short and it existed from 1929 to 1939. Based on the Latin script, a new Cyrillic alphabet was prepared in the 1940s of the last centuries. This writing was based on Russian graphics, which consists of 42 letters and considered the phonetic features of the Kazakh language. The current transition of the Kazakh language to Latin script is conditioned by the challenge of modernity, since after gaining independence many post-Soviet countries switched to Latin script.

In 2017, Kazakhstan approved the standard version of the Latin script of the Kazakh alphabet. According to its specifics, the new version of the Latin script of the Kazakh language is much closer to the Turkish language, which emphasizes the convergence of the cultures of the language, the historical culture of the Turkic peoples.

The current alphabet of the Latin script of the Kazakh language has 32 letters. To date, the transition to Romanization has already begun with basic documentation, the main documentation is produced in this writing. To facilitate the work of translating text, books, to raise the culture of the language, it is the phonetic expression of Kazakh letters, not allowing confusion of letters shown on Figure 1.

Әә - ***Á á***, Ғғ - ***Ǵ ǵ***, Ңң - ***Ń ń***, Өө - ***Ó ó***, Ұұ - ***Ú ú***, Үү - ***Ý ý***

Figure 1. Letters of the Kazakh language

This underlines the relevance of the chosen topic of recognition of Kazakh letters in Latin script. The Latin graphics of the Kazakh alphabet use 32 letters, which is shown in Figure 2.

Recognition of Latin letters of the Kazakh alphabet is an important task in the field of computer vision and image processing. Despite the fact that the Kazakh alphabet is based on the Latin alphabet, there are some differences, such as additional letters and punctuation marks.

Below we will provide reviews of research on the chosen topic in the work [1] of the authors of domestic scientists, a method for recognizing Latin letters of the Kazakh alphabet based on deep learning is proposed. The method is based on the use of a convolutional neural network, which is trained on a data set consisting of handwritten and printed letters.

Experimental results have shown that the proposed method provides high recognition accuracy.

ЖАҢА ҚАЗАҚ ӘЛІПБИІ

№	латынша	кириллше	әріптің атауы	№	латынша	кириллше	әріптің атауы
1	A a	А а	*a*	18	O o	О о	*o*
2	Ä ä	Әә	*ә*	19	Ö ö	Ө ө	*ө*
3	B b	Б б	*бы*	20	P p	П п	*пы*
4	D d	Д д	*ды*	21	Q q	Кк	*қы*
5	E e	Е е	*е*	22	R r	Р р	*ыр*
6	F f	Ф ф	*фы*	23	S s	С с	*сы*
7	G g	Г г	*гі*	24	Ş ş	Ш ш	*шы*
8	Ğ ğ	Fr	*ғы*	25	T t	Т т	*ты*
9	H h	Х х, Н h	*һы*	26	U u	У у	*ұу*
10	İ i	Й й, И и	*ій*	27	Ū ū	Ұ ұ	*ұ*
11	I ı	І і	*і*	28	Ü ü	Үү	*ү*
12	J j	Ж ж	*жы*	29	V v	В в	*вы*
13	K k	К к	*кі*	30	Y y	Ы ы	*ы*
14	L l	Л л	*ыл*	31	Z z	З з	*зы*
15	M m	М м	*мы*				
16	N n	Н н	*ны*				
17	D ŋ	Ңң	*ың*				

*C, X, W таңбалары цитаталық принциппен жазылатын шеттілдік сөздерде қолданылады.

Figure.2 Latin script of the Kazakh language

In [2], a method for recognizing Latin letters of the Kazakh alphabet based on machine learning is proposed. The method is based on the use of reference vector machines, which are trained on a data set consisting of handwritten and printed letters. Experimental results have shown that the proposed method provides good recognition accuracy.

The paper [3] proposes a method for recognizing Latin letters of the Kazakh alphabet based on statistical methods. The method is based on the use of probabilistic models that are trained on a data set consisting of handwritten and printed letters. Experimental results have shown that the proposed method provides satisfactory recognition accuracy.

Unlike the works of other authors, this article presents algorithms constructed by the gradient descent method and analyses of the results obtained.

All these articles are an overview of various methods. Each of the methods has its advantages and disadvantages. Methods based on deep learning provide high recognition accuracy, but require large computational resources. Methods based on machine learning provide good recognition accuracy and require less computing resources than methods based on deep learning.

2. Materials and methods

To develop a computational algorithm for recognizing Kazakh letters of the Latin alphabet, a model is introduced according to which the recognition algorithm will be trained, the model uses operations on labels introduced by us for letters, for predictions of the target variable, that is, there is a mapping from the space of labels to the space of target predictions [9]:

$a: X \to Y$

where $a \in A$ is a family of models. Next, rewriting the model in the form - y , where y is in turn equal to $y = a(x_i, w, h)$

Where x_i - is the label vector for the *i*-th letter, *w*- are the model parameters (optimized by the model algorithm) h are the hyperparameters of the model (optimized by those who run machine learning algorithms). After the model is selected, we begin to train it by dividing the training of the model into training and test samples.

A training sample - is a data set for which we know the «letter models – target variable» for each letter from the sample.

A test sample - is a data set for which only the labels of the letter model are known. In order to assess how bad or good a given model is, we use the loss function $L(y_i, y_i)$ to assess the quality of the model. For a specific letter $y_i - y_i$ the prediction of the model and the real value of the target variable coincide, then the loss functions $L(y_i, y_i)$ takes small values. If the prediction of the model and the real value differ, then the loss function takes on large values. Using the data from the training sample, we estimate the loss functional. The loss functional is the average value of the loss function for all labels from the training sample:

$$Q(a, X) = \frac{1}{n} \sum_{i=1}^{n} L(y_i, a(x_i, w, h))$$

Therefore, the purpose of the training will be as follows

$$Q(a, X) \to \min_{a \in A}$$

where $a \in A$ a family of models.During the training process, we must select such parameters and hyperparameters of the model that best predict the target values in the training sample.

In the learning process, there is a certain pattern in which simple models contain a limited number of features and the dependence between variables have large values of loss functions, complex models that have many features and there are complex dependencies between variables may have low values of loss functions. When training the model, the loss functions on the test data may initially decrease, but at some point a situation may arise when the losses on the test sample begin to grow again, and the losses on the training sample continue to fall. Using an approach called validation in this case, we take a «piece» of the training sample, postponing it, train the model on the rest of the training sample and test it on the deferred «piece». This type of validation is called the use of deferred sampling [9] (train_test_split), Figure 3.

Figure 3. Deferred sampling scheme

Preparing data for machine learning models using TensorFlow and Keras frameworks.

The model learning algorithm is described below:

1. Import libraries**:** The necessary modules and functions are imported from **TensorFlow, Keras and Scikit-Learn.** This includes layers and models from Keras, a utility for converting labels to a categorical format, functions for dividing data into training and test samples, as well as a module for preprocessing.

2. Image preprocessing**:** The **preprocess_images_with_opencv** function, defined earlier, is called to convert a set of images and their labels into a format suitable for training the model. The result is stored in the variables **preprocessed_data and preprocessed_labels.**

3. Preparation of input data and labels: **X** is initialized as a **preprocessed_data** containing images. **y** is initialized as **preprocessed_labels** containing labels.

4. Encoding of labels: An instance of LabelEncoder is created to convert labels from text format to numeric format. Labels are encoded using the **fit** and **transform** method of this encoder.

5**.** Converting labels to one-hot format: The **to_categorical** function from Keras is used to convert numeric labels to **one-hot encoding** format. This is a common practice when processing categorical data for classification tasks.

The comment indicates that 60 classes are assumed (there is a slight inaccuracy in the comment mentioning 26 classes, which may be a discrepancy with the actual number of classes in this case). This code prepares data for neural network training by processing images and converting labels into a format suitable for classification models in **TensorFlow/Keras.**

Importing the Tensorflow framework, Keras and their components, as well as functions for dividing data into training and validation samples, we divide the data into training and validation samples (Figure 2.) while using the train_test_split function from the Scikit-Learn library to separate data (X) and labels (y) into training (X_train, y_train) and validation (X_val, y_val) samples. 20% of the data is allocated for validation (test_size=0.2).

Next, repeating the process of dividing the sample into training and validation, we use n-Fold cross-validation. The training sample is divided into n parts of the same volume, which contain different objects. N iterations are performed and at each iteration the model is trained on (n-1) parts of the training sample, and the model is also tested on a training sample that did not participate in the training.

The basic Keras data structure is a model, a way of organizing layers. There are two basic types of models available in Keras: the Sequential model and the Model class used with the functional API. We used the Sequential model, which creates CNN models using a sequential stack of layers (models Sequential).

Convolutional layers are the main block of a convolutional non–linear network. The convolution layer includes its own filter for each channel, the convolution core, which processes the previous layer in fragments (summing up the results of the matrix product for each fragment). The weight coefficients of the convolution core (a small matrix) are unknown and are set during the learning process. The peculiarity of convolutional layers is the relatively small number of parameters set during training.

Conv2D - is a 2D mesh layer (for example, spatial convolution over images). This layer creates a convolution core to create an output tensor. The model includes several convolutional layers **(Conv2D**) with **ReLU** activation and pooling layers **(MaxPooling2D)**, as well as fully connected layers (**Dense**) and a regularization layer (**Dropout**). The output layer uses the **softmax** activation function to classify images into several classes. The method of configuring the model for training includes the compilation optimizer **'adam'**, the loss function 'categorical_crossentropy' and the metric **'accuracy'**. The model is compiled with the **'adam'** optimizer, the 'categorical_crossentropy' loss function (suitable for multiclass classification) and the **'accuracy'** metric.

The model is trained on X_train and y_train data indicating the number of epochs (epochs), batch size (batch_size) and validation data (X_val, y_val). The number of epochs and batch size can be adjusted depending on the specifics of the task and the capabilities of the computing system.

The results obtained: The model will be trained on the prepared data, and then evaluated based on its performance on the validation dataset (Figure 4).

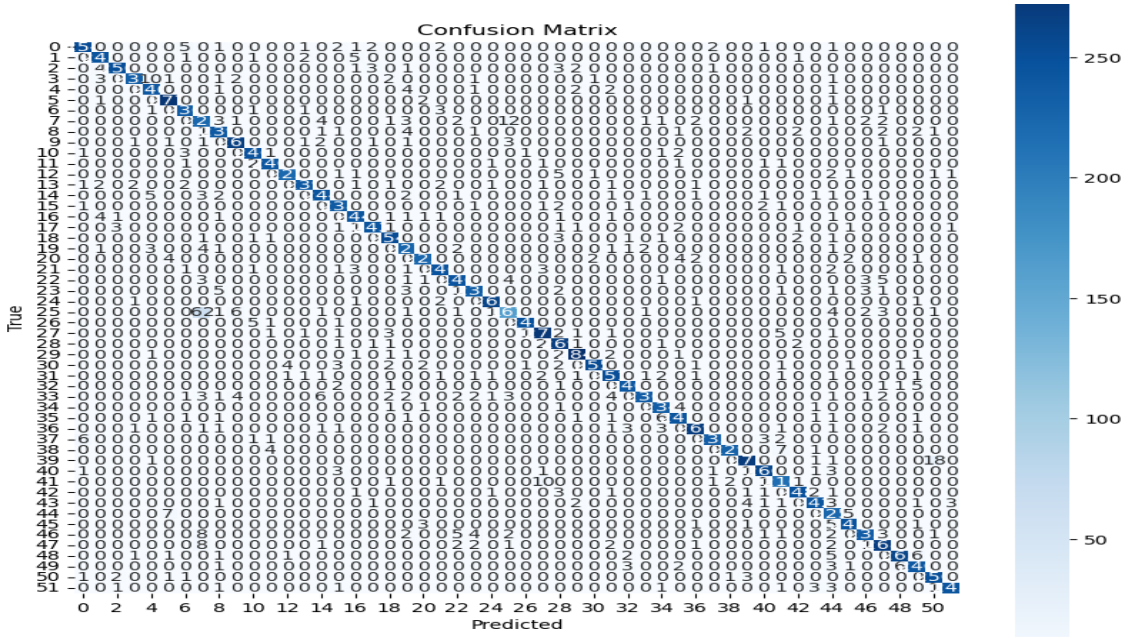

Figure 4. Convolutional neural network (CNN) training using Tenzorflow and Keras for image classification

*Analysis of the results obtained***.** The analysis of the error matrix shows the following:

- The overall accuracy of the model is approximately 93.90%. This means that about 93.90% of the model's predictions were correct.

- Accuracy by class (Precision), Completeness (Recall) and F1-measure (F1 Score) for each class show different levels of performance. These metrics are useful for determining how well the model works for each specific class.

For example: For a class with index 0, accuracy is 95.85%, completeness is 93.38%, and F1 measure is 94.60%.

For a class with an index of 25, accuracy is 86.56%, completeness is 65.71%, and F1-measure is 74.71%. This may indicate that the model is having difficulty correctly recognizing this class.

These metrics provide valuable information about the performance of the model and can be used to further improve and customize the model (Figure 5).

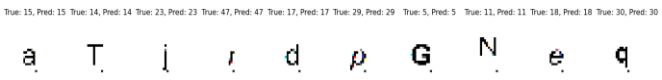

Figure 5. The model

Analyzing the visualization of tsn, several observations can be made regarding the distribution (Figure 6).

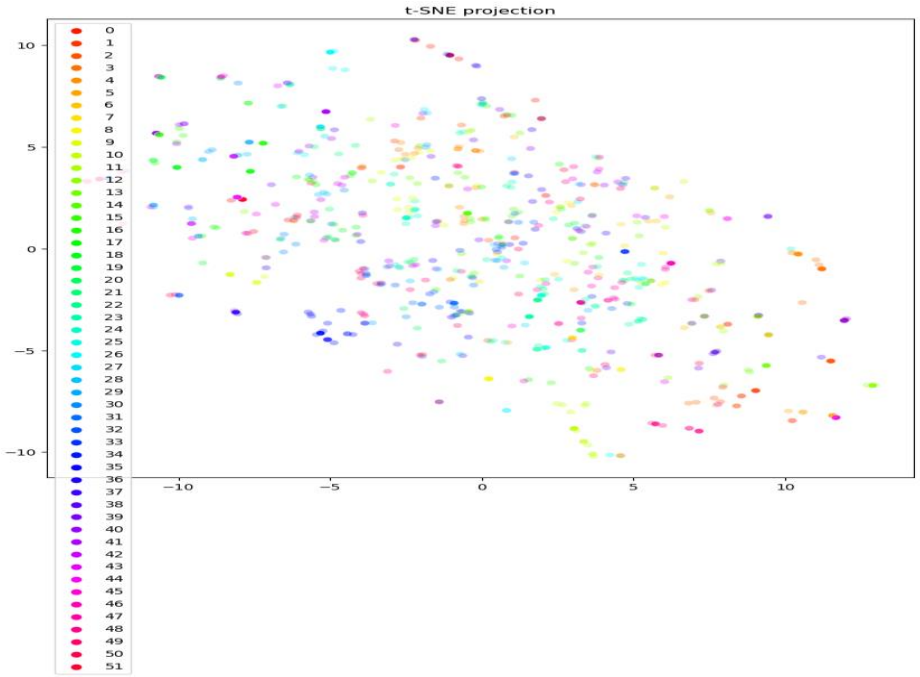

Figure 6. Analysis of tSNE visualizations

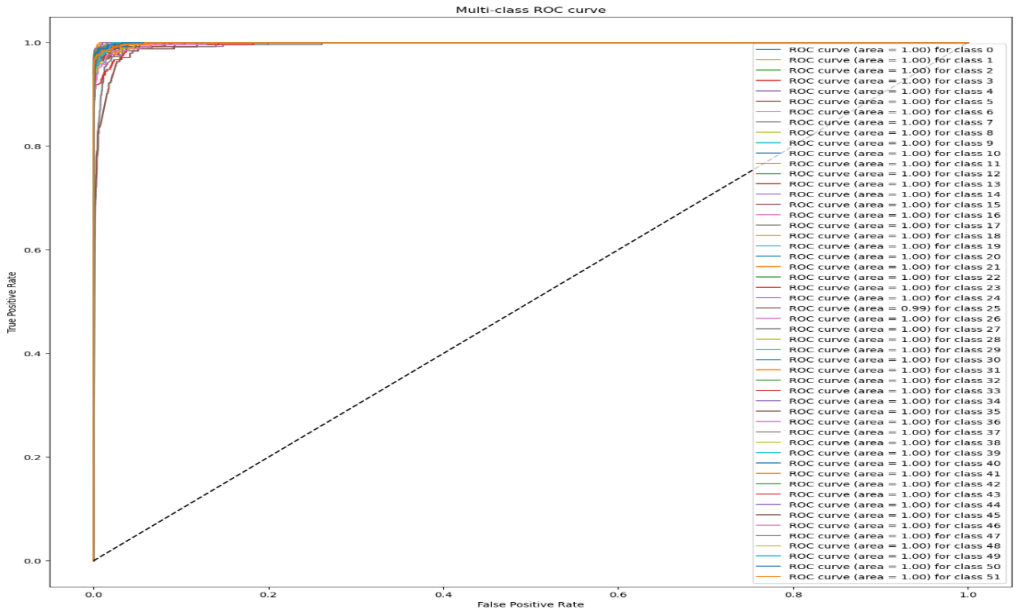

Figure 7. Multiclass ROC curve (model representations at all classification thresholds)

By analyzing the visualization of t-SNE, several observations can be made regarding the distribution and structure of the data:

1. Data clustering: t-SNE has effectively mapped multidimensional data into a two-dimensional space in such a way that the data is grouped into clusters. Clusters with the same color represent data of the same class. Well-defined and isolated clusters can indicate good class distinctiveness.

2. Overlap between clusters: Some clusters show overlap or proximity to other clusters. This overlap may indicate the likelihood of classification errors, since the model may mistakenly assign data from one cluster to a neighboring one.

This is especially noticeable for clusters in the central part of the graph.

3. Outliers: The graph shows points that do not belong to explicit clusters or are far from their main groups. These outliers may represent abnormal data or errors in the data that may make classification difficult.

4. Cluster density: Some clusters are denser and more compact, indicating high consistency of data within the class. While more scattered clusters may indicate intra-class variability.

5. Class homogeneity: If clusters are distinct, separate groups, this may indicate homogeneity within classes. If the clusters are blurred and fuzzy, this may indicate heterogeneity or class mixing.

Figure 7 below shows a multiclass ROC curve (Receiver Operating Characteristic curve), which is a graphical representation of the performance of the classification model at all classification thresholds.

3. Results and discussion

Figure 6 shows a multiclass ROC curve (Receiver Operating Characteristic curve), which is a graphical representation of the performance of the classification model at all classification thresholds. The main aspects of this graph are:

Each line of the ROC curves corresponds to one class. The ideal model will have a ROC curve going straight up the sensitivity axis (True Positive Rate) and then to the right along the specificity axis (1 - False Positive Rate), which means that the model has a perfect difference between classes.

The area under the curve (AUC) close to 1.00 indicates a very high performance of the model in the classification of this class. Ideally, the AUC should be as close to 1 as possible. In this graph, for almost all classes, the AUC is 1.00, which indicates an exceptionally high classification accuracy.

The diagonal dotted line represents a random guess. Any ROC curve above this line indicates better performance than random guessing.

The ROC curve for class 25 has an AUC lower than 1.00 (the exact value is not visible, but it is indicated that it is less than 1). This indicates that the classification for this class is not ideal and there is some room for error.

The performance of the model as a whole indicates a high performance of the classification model, since most classes have an AUC of 1.00. This means that the model separates positive and negative cases very well for most classes.

It is important to note that although the ROC curve can provide performance information for all thresholds, in practice it is also important to look at other metrics such as accuracy, completeness, and F1 measure to get a complete picture of the model's performance.

4. Conclusions

Recognition of Latin letters of the Kazakh alphabet is an important task in the field of computer vision and image processing. The gradient descent method was used in the study. The Kazakh alphabet, based on Latin, has some differences, such as additional letters and punctuation marks.

The paper provides an overview of existing recognition methods, including methods based on machine learning. The algorithm proposed in the article for recognizing Latin letters of the Kazakh alphabet can be used in various applications, such as optical character recognition (OCR) systems, automated verification of entered texts using mobile devices.

The algorithm uses operations on labels entered for letters to predict the target variable. The experiments have shown that the proposed algorithm provides high accuracy of recognition of letters of the Latin Kazakh alphabet.

Acknowledgements

We studied this study independently under the guidance of the teacher Ph.D. Pirniyazova. We express our great gratitude to our teacher for the support and direction in the study.

References

[1] Abdrakhmanov, A.K., Bekbosynov, B.K. (2023). Recognition of Latin letters of the Kazakh alphabet based on deep learning. *Bulletin of KazNTU, 17(3), 123-128.*

[2] Amanzholova, A.T., Iskakova, A.K. & Kudaibergenov, D.T. (2022). Recognition of Latin letters of the Kazakh alphabet based on machine learning. *Bulletin of KazNTU, 16(2), 120-125*

[3] Eskendirov, A.B., Abdrakhmanov, A.K., Bekbosynov, B.K. (2021). Recognition of Latin letters of the Kazakh alphabet based on statistical methods. *Bulletin of KazNTU, 15(1), 116-121*

[4] Katılmış, Z. & Karakuzu, C. (2021). Journal Pre-proofs ELM Based Two-Handed Dynamic Turkish Sign Language (TSL) Word Recognition. *Expert Systems with Applications, (182), 115213*. https://doi.org/10.1016/j.eswa.2021.115213

[5] Sir Eiad Almekhlafi, Moeen AL-Makhlafi, Erlei Zhang, Jun Wang, Jinye Peng. (2022). A classification benchmark for Arabic alphabet phonemes with diacritics in deep neural networks. *Computer Speech & Language, (71), 101274.* https://doi.org/10.1016/j.csl.2021.101274

[6] Somsap, S., Seresangtakul, P. (2020). Isarn Dharma Word Segmentation Using a Statistical Approach with Named Entity Recognition. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP), 19(1), 1-16.* https://doi.org/10.1145/3359990

[7] Danilov, A.V., Zaripova, R.R., Salekhova, L.L. (2017). The application of statistical methods in the development of Cyrillic-Latin converter for Tatar language. *Journal of Fundamental and Applied Science, 9(7S)*

[8] Jyoti Pareek, Dimple Singhania, Rashmi Rekha Kumari, Suchit Purohit. (2020). Gujarati Handwritten Character Recognition from Text Images. *Procedia Computer Science, (171), 514–523.* https://doi.org/10.1016/j.procs.2020.04.055

[9] Gafarov, F.M., Galimyanov, A.F. (2018). Artificial neural networks and applications: studies. the manual. *Kazan: Kazan Publishing House*

[10] Dolganov, A.Yu., Ronkin, M.V. & Sozykin, A.V. (2023). Basic machine learning algorithms in Python: textbook. *Yekaterinburg: Publishing House of the Ural University*

[11] Pirniazova, P.M., Son, E.Yu. & Kulzhan, D.J. (2023). Copyright certificate No. 41683 for the program «Recognition of Latin letters of the Kazakh alphabet based on a neural network»

Нейрондық желі негізінде қазақ әліпбинің латын әріптерін тану

П.М. Пирниязова*, Е.Ю.Сон, Д.Ж. Құлжан

Satbayev University, Алматы, Қазақстан

**Корреспонденция үшін автор: pirniyazova1974@gmail.com*

Аңдатпа. Бұл мақалада TensorFlow фреймворкін пайдалана отырып, нейрондық желі негізінде қазақ әліпбиінің латын әріптерін тану қарастырылады. Ол үшін turtle графикалық кітапханасының көмегімен латын әліпбиінің қазақ әріптерінің дерекқоры құрылады. Алгоритм әріптер үшін еңгізілген таңбалар үстінде операцияларды қолданады, мақсатты айнымалыны болжау үшін. Жүргізілген эксперименттер ұсынылған алгоритм латын қазақ әліпбиінің әріптерін танудың жоғары дәлдігін қамтамасыз ететіндігін көрсетті. Латын әліпбиінің қазақ әріптерін танудың есептеу алгоритмі әзірленді. Бағдарламалық жасақтама латын әліпбиінің әріптерін таниды. Алынған нәтижелер графикалық түрде суреттелген.

Негізгі сөздер: *тану, нейрондық желілер, перцептрон, синапс, салмақ матрицасы, пиксель матрицасы, қабаттар, қателер*.

Распознавание латинских букв казахского алфавита на основе нейронной сети

П.М. Пирниязова*, Е.Ю. Сон, Д.Ж. Құлжан

Satbayev University, Алматы, Казахстан

**Автор для корреспонденции: pirniyazova1974@gmail.com*

Аннотация. В данной статье рассматривается распознавание латинских букв казахского алфавита на основе нейронной сети с использованием фреймворка TensorFlow. Для этого с помощью графической библиотеки turtle создается база данных казахских букв латинского алфавита. Алгоритм использует операции над вставленными символами для букв, чтобы предсказать целевую переменную. Проведенные эксперименты показали, что предложенный алгоритм обеспечивает высокую точность распознавания букв латинского казахского алфавита. Разработан вычислительный алгоритм распознавания казахских букв латинского алфавита. Программное обеспечение распознает буквы латинского алфавита. Полученные результаты проиллюстрированы графически.

Ключевые слова: *распознавание, нейронные сети, перцептрон, синапс, весовая матрица, матрица пикселей, слои, погрешности*.

Received: 02 June 2023
Accepted: 15 September 2023
Available online: 30 September 2023