DOI: 10.37943/15LPCU4095

Samat Mukhanov*
Master of Technical Sciences, senior-lecturer, Department of Computer Engineering
s.mukhanov@iitu.edu.kz, orcid.org/0000-0001-8761-4272
International Information Technology University, Almaty, Kazakhstan
Raissa Uskenbayeva
Doctor of technical science, professor, Head of Institute of Automation, and Information Technologies
r.k.uskenbayeva@satbayev.university, orcid.org/0000-0002-8499-2101
Satbayev University, Almaty, Kazakhstan
Young Im Cho
PhD, professor
yicho@gachon.ac.kr, orcid.org/0000-0003-0184-7599
Faculty of Computer Engineering Gachon University, Seoul, South Korea
Kabyl Dauren
Master student
d_kabyl@kbtu.kz, orcid.org/0009-0005-4837-8728
Kazakh British Technical University, Almaty, Kazakhstan
Les Nurzhan
Master student
38530@iitu.edu.kz, orcid.org/0009-0008-2909-3606
International Information Technology University, Almaty, Kazakhstan
Maqsat Amangeldi
Master student
38517@iitu.edu.kz, orcid.org/0009-0002-0899-2975
International Information Technology University, Almaty, Kazakhstan

GESTURE RECOGNITION OF MACHINE LEARNING AND CONVOLUTIONAL NEURAL NETWORK METHODS FOR KAZAKH SIGN LANGUAGE

Abstract: Recently, there has been a growing interest in machine learning and neural networks among the public, largely due to advancements in technology which have led to improved methods of computer recognition of objects, sounds, texts, and other data types. As a result, human-computer interactions are becoming more natural and comprehensible to the average person. The progress in computer vision has enabled the use of increasingly sophisticated models for object recognition in images and videos, which can also be applied to recognize hand gestures. In this research, popular hand gesture recognition models, such as the Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), and Support Vector Machine (SVM) were examined. These models vary in their approaches, processing time, and training data size. The important feature of this research work is the use of various machine learning algorithms and methods such as CNN, LSTM, and SVM. Experiments showed different results when training neural networks for sign language recognition in the Kazakh sign language based on the dactyl alphabet. This article provides a detailed description of each method, their respective purposes, and effectiveness in terms of performance and training. Numerous experimental results were recorded in a table, demonstrating the accuracy of rec-

Received: 31.07.2023 Accepted: 13.09.2023 Published: 30.09.2023

Scientific Journal of Astana IT University
ISSN (P): 2707-9031 ISSN (E): 2707-904X
VOLUME 15, SEPTEMBER 2023

ognizing each gesture. Additionally, specific hand gestures were isolated for testing in front of the camera to recognize the gesture and display the result on the screen. An important feature was the use of mathematical formulas and functions to explain the working principle of the machine learning algorithm, as well as the logical scheme and structure of the LSTM algorithm.

Keywords: Hand gesture recognition; neural networks; CNN; LSTM; SVM.

Introduction

At present, Kazakhstan has a population of over 200,000 individuals who are unable to speak and more than 80,000 who are hearing-impaired. On a global scale, there are approximately 430 million people facing hearing impairment, with 70 million experiencing complete deafness [1]. This issue poses a daily challenge for these individuals, as they often encounter difficulties in communication and rely on interpreters for interaction. Unfortunately, many of them cannot afford solutions to their problems, such as hearing aids, which can cost up to an average of 500,000 tenge and involve ongoing replacement expenses. Given the scale of this problem, the responsibility of seeking ways to assist the deaf and mute has been taken. Various approaches to aid these individuals have been explored, with their primary struggle being the inability to communicate, comprehend, or receive support from those around them. The Kazakh sign language is as unique as sign languages in other countries. The Kazakh tactile alphabet consists of 42 letters, which means there are the same number of hand gestures with different hand positions.

GoogleAI presents a method for precise tracking of hands and fingers through machine learning within the MediaPipe cross-platform framework, which also encompasses data processing pipelines handling video, audio, and time series data [2]. The authors uses models such as the palm detector (blazePalm), a model for identifying key points on the hand, and a gesture recognition algorithm. These models collectively constitute the foundation of the mentioned framework. Each model is distinct and plays a crucial role in recognizing specific elements in gesture recognition.

Literature review and problem statement

In this work, the algorithms and methods of machine learning in a convolutional neural network for the recognition of gestures in the Kazakh sign language will be used. Consequently, all these algorithms and provide detailed descriptions of how they work will be explored and tested. Convolutional neural networks are frequently and effectively employed in deep learning, particularly in image recognition. The training of this network is based on machine learning and constitutes not only an interconnected neural network but also what it consists of and what metrics it employs for predictions in pattern recognition, in our case, for recognizing gestures in the Kazakh sign language. This work will showcase the functions and mathematical formulas used by this convolutional network [3]. Convolutional Neural Networks (CNNs) are a type of deep neural network that is specifically designed for image and video processing tasks. CNNs consider the spatial relationships between pixels and apply convolutional operations to extract increasingly complex features, starting from simple edges and progressing to more intricate objects. These networks have demonstrated outstanding performance on a variety of computer vision tasks and have shown promising results in other domains as well. Yann LeCun first introduced CNNs in the 1980s, but they did not become mainstream until the mid-2010s with the development of deep learning techniques and the availability of large datasets. Today, CNNs are widely used in both industry and academia and are expected to play an increasingly critical role in emerging technologies.

DOI: 10.37943/15LPCU4095
© Samat Mukhanov, Raissa Uskenbayeva,
Young Im Cho, Kabyl Dauren, Les Nurzhan,
Maqsat Amangeldi

The CNN architecture is composed of Convolutional, Pooling, and Fully Connected layers. The foundational component of CNNs is the convolutional layer, which extracts features from the input data. This layer comprises learnable filters or kernels, which are applied to the input data using convolutional operations to produce a set of feature maps. The convolutional operation involves multiplying the kernel values with the corresponding input values and then summing the products to obtain a value that is placed into the corresponding position of the output feature map. This operation is performed for each location in the input data, resulting in a set of output feature maps. The mathematical operation of two functions (Convolutional operation):

$$[f * g](t) \equiv \int_0^t f(\tau)g(t-\tau)d\tau\ , (1) \qquad (1)$$

The convolutional operation is a fundamental part of convolutional neural networks. It involves sliding a small kernel over the input data and computing the dot product between the kernel and the local region of the input that it is currently covering. This process is repeated for each location in the input data, producing a feature map that represents the presence of specific features or patterns in the data. The convolutional operation is closely related to the concept of cross-correlation, but with a key difference: before being applied to the input data in cross-correlation, the kernel is flipped both horizontally and vertically. Despite this difference, the two operations are often used interchangeably in the context of convolutional neural networks.

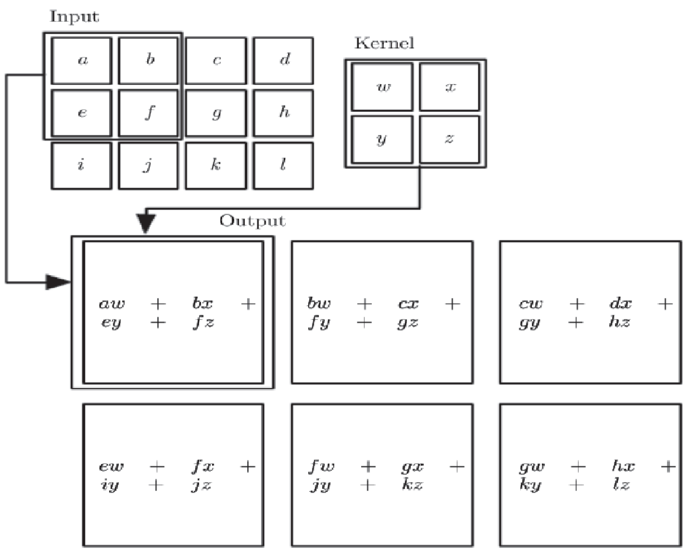


Figure 1. An example of 2-D convolution.

The number of kernels in the layer affects how many output feature maps are produced. Each kernel generates one output feature map, which captures a specific aspect of the input data. The task's complexity and the characteristics of the input data determine the kernel size and the quantity of output feature maps. Each output feature map is a two-dimensional array that corresponds to a specific filter and captures the presence of that filter in the input data. The filters are learned through backpropagation during the training process, with the objective of maximizing the performance of the model on the task at hand.

In addition to the kernels, the convolutional layer may also include channels, which are used to capture different aspects of the input data. Each channel is a two-dimensional array that represents a specific feature of the input data. As an example, in a picture classification task, distinct channels for capturing edges, textures, and colors may be present in the input data. The combination of channels and kernels enables sparse interactions, parameter sharing, and equivariant representations, which are key concepts in the design of CNNs. Sparse interactions refer to the fact that each output feature map is generated by convolving a small kernel with a subset of the input feature maps, rather than with all input feature maps. This increases the effectiveness of the model and lowers the number of parameters required to represent the layer.

Parameter sharing refers to the fact that the same kernel is used to generate each output feature map in a convolutional layer. This means that the weights of the kernel are shared across all output feature maps, rather than being learned independently for each feature map. Parameter sharing reduces the number of parameters needed to represent the model and helps to improve its generalization performance.

The authors in [4, 5] citations introduce a solution designed to improve the recognition of surgical hand gestures by employing a capsule network in a contactless surgical environment. This work specifically addresses contactless sign language recognition as a means of reducing infection risks and enhancing sign language recognition systems. The research underscores the superior efficiency of Capsule Networks (CapsNets) when compared to alternative methods. The approach entails the utilization of a CapsNet in conjunction with Leap Motion to extract and preprocess infrared images at a rapid rate of 60 frames per second. The process includes training diverse network models and assessing gesture recognition within the surgical setting. The application of the CapsNet method achieves an impressive classification accuracy of 86.46%, surpassing the 73.67% achieved by a Convolutional Neural Network (CNN).

Methods and materials

The method of Support Vector Machines (SVM)

One of the popular machine learning methods, Support Vector Machine (SVM), is often used in classification and regression analysis tasks. SVM belongs to the family of supervised learning algorithms, which means that it requires labeled training data for its training. The main goal of SVM is to determine a hyperplane that effectively separates data points into different categories. The hyperplane is defined as the boundary that maximizes the margin between classes. The margin in SVM is defined as the distance between the hyperplane and the nearest data points from each class. The SVM algorithm finds the hyperplane that maximizes this margin, leading to better generalization on new data and improved classification performance [6]. SVMs can be used for classification tasks, where the goal is to determine the optimal hyperplane that can separate data points into different classes, as well as for regression tasks, where the goal is to predict continuous values. In the case of binary classification, the SVM algorithm aims to find a hyperplane that can maximize the margin between two classes, determined by the distance between the nearest points of each class and the hyperplane. Support Vector Machines (SVM) can be classified into two main types: linear SVM and non-linear SVM. Linear SVM is a type of Support Vector Machine method that uses linear hyperplanes to separate data into classes. It is used for classification tasks where the data is linearly separable, meaning the data can be divided into two different classes using a straight line. The main idea of linear SVM is to find the best hyperplane that maximizes the margin between two classes. The margin is the distance between the nearest data points from each class and the hyperplane. Support vectors, which are data points closest to the hyperplane, are of great importance in the SVM method. The hyperplane is used to maximize the margin when separating two classes. This is achieved by determining how to maximize the margin while minimizing the classification error in the optimization problem. There are several methods to solve the

DOI: 10.37943/15LPCU4095
© Samat Mukhanov, Raissa Uskenbayeva, Young Im Cho, Kabyl Dauren, Les Nurzhan, Maqsat Amangeldi

optimization problem, including quadratic programming methods. The scheme is described below for the linear SVM:

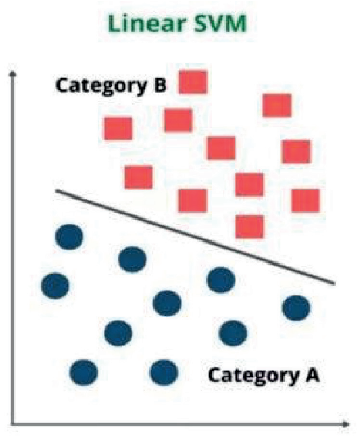


Figure 2. Linear SVM schema

Nonlinear SVMs are more powerful than linear SVMs because they can handle data that cannot be divided linearly.The scheme is described below for the Nonlinear SVM:

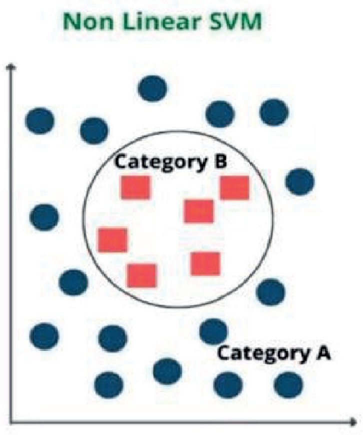


Figure 3. Nonlinear SVM scheme

The CNN method in recognition problems

In this research work Convolutional Neural Network for training machine learning methods for hand gesture and Kazakh sign language recognition was used. The results of recognition have been tested in this work. Firstly, it is necessary to introduce this neural network and how it works in pattern recognition. Convolutional Neural Networks (CNNs) are a specific type of artificial neural network that is highly effective at processing and analysing visual data such as images and videos. The basic architecture of CNNs involves passing the input data, such as an image, through interconnected layers of artificial neurons. Each neuron in a layer is connected to a small region of the input data, and these regions overlap with one another to cover the entire input.

The convolutional layer is a critical component of CNNs and uses learnable filters called kernels to extract features from the input data. During the process of convolution, the kernel slides over the input data and multiplies its values with the corresponding input values, resulting in a set of feature maps. The number of output feature maps is determined by the

Scientific Journal of Astana IT University
ISSN (P): 2707-9031 ISSN (E): 2707-904X
VOLUME 15, SEPTEMBER 2023

number of kernels in the layer, with each kernel generating one output feature map that captures a specific aspect of the input data. The combination of channels and kernels allows for sparse interactions, parameter sharing, and equivariant representations, which are important concepts in the design of CNNs [7]. These concepts contribute to improving the efficiency and generalization performance of the machine learning system.

The Recurrent Neural Network method Long Short-Term Memory

Specifically, this method is applied in speech recognition because it accurately and effectively analyses sound frequencies and temporal sequences. However, the method also performs well in pattern recognition, particularly in recognizing gestures. The application of LSTM in this manner demonstrates its effectiveness in handling complex dynamic hand gesture recognition and correctly predicting which symbol or letter is depicted in the image captured by the webcam. All letters from the Kazakh sign language tactile alphabet were utilized in the training and testing processes. LSTM is a technique used in recognizing gestures that involves identifying gestures across multiple frames. To accurately identify dynamic movements, hand movements need to be emphasized across several frames.

The LSTM architecture involves a memory cell that contains gate units which perform operations on incoming data and store the result in the cell's state and hidden state. Initially, Hochreiter and Schmidhuber used two types of gates: input and output gates. Later, in 2000, Gers et al. added a forget gate to their work, resulting in a memory cell model with three gates that is now widely used in LSTM [8].

In Figure 1, it is seen that the memory cell has two input paths that receive information about the state and hidden state of the previous cell, and two output paths that send the current state and hidden state to the next memory cell.

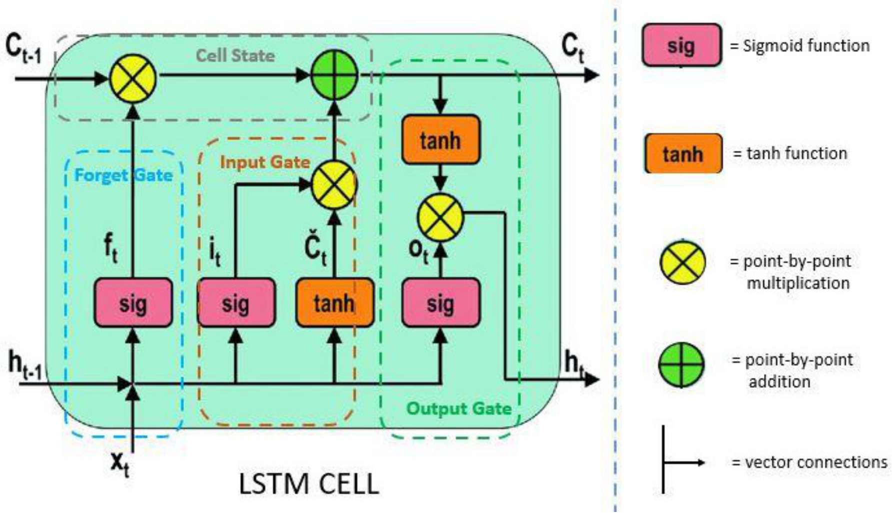


Figure 4. LSTM's memory cell

The sigmoid function is assigned to each gate unit, which restricts their output value between 0 and 1. The forget gate is the first unit in the memory cell, which evaluates whether the previous state should be forgotten or retained. If the output is 0, then all the previous information will be forgotten, while an output of 1 will keep all the previous information. The neural network's ability to forget unnecessary data improves its flexibility and results in fewer errors during the training process. The forget gate function is shown below:

$$f_t = \sigma\left(x_t * U_f + h_{t-1} * W_f\right), \tag{2}$$

here:
x_t – input value of the current timestamp
U_f – forget weight of the current input value
$h_{(t-1)}$ – the hidden state of the cell at previous timestamp
W_f – forget weight associated with hidden state

The function of the input gate is to decide whether new information from the current timestamp should be added to the cell's state. It works in tandem with the new information and determines whether it should be written or not. If the input gate value is 0, the new information is not remembered, while a value of 1 means that all new information will be remembered.:

$$i_t = \sigma(x_t * U_i + h_{t-1} * W_i), \tag{3}$$

here:
x_t – input value of the current timestamp
U_i – input weight of the current input value
$h_{(t-1)}$ – the hidden state of the cell at previous timestamp
W_i – input weight associated with hidden state

The function for new information has a tangent value that produces an output in the range of -1 to 1. A negative value between -1 to 0 indicates that the new information is less significant, while a value between 0 to 1 indicates that the new information carries essential data:

$$N_t = tanh(x_t * U_c + h_{t-1} * W_c), \tag{4}$$

here:
x_t – input value of the current timestamp
U_c – weight of the current cell state
$h_{(t-1)}$ – the hidden state of the cell at previous timestamp
W_c – weight of the current hidden state

Thus, you can determine the current state of the cell using the following formula:

$$C_t = f_t * C_{t-1} + i_t * N_t, \tag{5}$$

here:
f_t – forget gate value
$C_{(t-1)}$ – the state of the cell at previous timestamp
i_t – input gate value
N_t – new information value

The output gate does not modify the state of the cell but determines which information from the cell state should be included in the output. The output gate function has a sigmoid value between 0 and 1, where 0 means no contribution to the output, and 1 means full contribution to the output:

$$o_t = \sigma(x_t * U_o + h_{t-1} * W_o), \tag{6}$$

here:
x_t – input value of the current timestamp
U_o – output weight of the current input value
$h_{(t-1)}$ – the hidden state of the cell at previous timestamp
W_o – output weight associated with hidden state

The hidden state of a cell is determined by the following formula:

$$h_t = o_t * tanh(C||t), \tag{7}$$

Further, all these steps will be repeated for each data sequence.

LSTMs have found application in various domains such as language modeling where they have been utilized for text generation, predicting the next word in a sentence, and language translation. They have also been employed in speech recognition to convert spoken words into text. In addition, LSTMs have been used in image captioning to generate descriptions for images, which can be beneficial for individuals with visual impairments. They have also been applied to video analysis tasks such as action recognition and video captioning. Finally, LSTMs have been used in time series analysis to forecast stock prices, weather patterns, and other time series data.

Comparative analysis of Machine Learning and Neural Network methods

Authors [9] compare machine learning methods in their work. The research compares various machine learning algorithms, including KNN, Decision Tree, Random Forest, XGBoost, and LightGBM, with a particular emphasis on Random Forest, which yielded the highest model score. The study also employs wavelet-transformed data to identify geological properties.

In our case, the LSTM proved to be the most suitable choice since it excels in recognizing dynamic hand gestures. Although initially widely used in speech recognition, numerous authors in various scientific articles have demonstrated how they applied this algorithm to their own experiments and published their findings in various publications and high-impact journals. The performance of CNNs, LSTMs, and SVMs can also vary depending on conditions. In situations where the data set is small or the problem is with few training examples, SVMs may perform better due to their ability to generalize well on limited data. CNNs and LSTMs may struggle to generalize from limited data and may require a significant amount of training data to achieve good results.

In cases where the data is highly consistent, such as in time series analysis, LSTMs can outperform CNNs and SVMs due to their ability to capture long-term dependencies in data.

For tasks with high-dimensional data, such as image classification, CNNs can be the most efficient due to their ability to capture spatial patterns in the data. SVMs can also work well with multidimensional data but may require more processing time and computational resources. In situations where the dataset contains noisy or irrelevant features, SVMs can be the most efficient due to their ability to handle high-dimensional data with fewer irrelevant features. CNN and LSTM may be more prone to overfitting in such scenarios. Therefore, the choice of CNN, LSTM or SVM in practice depends on the specific conditions of the problem under consideration, how big the data set is, the nature of the data, and the presence of noisy or irrelevant features. Then we will compare in practice and choose the best one for our purpose.

Leap Motion is a hand gesture recognition database, comprising a collection of images captured in the near-infrared spectrum using the Leap Motion sensor [10]. This database encompasses 10 distinct hand gestures executed by 10 different individuals (comprising 5 men and 5 women). The dataset is organized into separate folders and extends beyond gesture recognition to include sign language recognition. The primary objective is to address a social need, allowing individuals with hearing impairments to communicate with minimal limitations. Contemporary technologies have the potential to effectively address this challenge.

The authors of reference [11] have introduced a novel system designed for communication with individuals who have visual impairments or rely on radio communication. The study employed an experimental approach, demonstrating that the algorithmic device can successfully identify icons within video sequences featuring minimal clutter and dynamic backgrounds, thanks to CapsNet technologies. Their methodology incorporated HSV Segmentation and Finger-tip detection algorithms, and they also employed Support Vector Machines for sign language recognition. As a result, they achieved an impressive 93% accuracy in recognizing static gestures using the HSV Segmentation and Finger-tip detection methods. Additionally, the authors conducted comparisons with established methods, presenting the outcomes in a tabulated format.

Architecture and models

An automated system has been proposed to recognize hand gestures for specific letters using biorthogonal wavelet transformation. The system involves several steps: first, the images are read and filtered to remove noise. Then, the edges of the images are detected, and projections along specific directions are calculated using the Radon transform. Next, biorthogonal wavelet transformation is applied to these projections. The system uses SVM to train and test the recognition of hand gestures. Figure 5 provides an overview of the entire system.

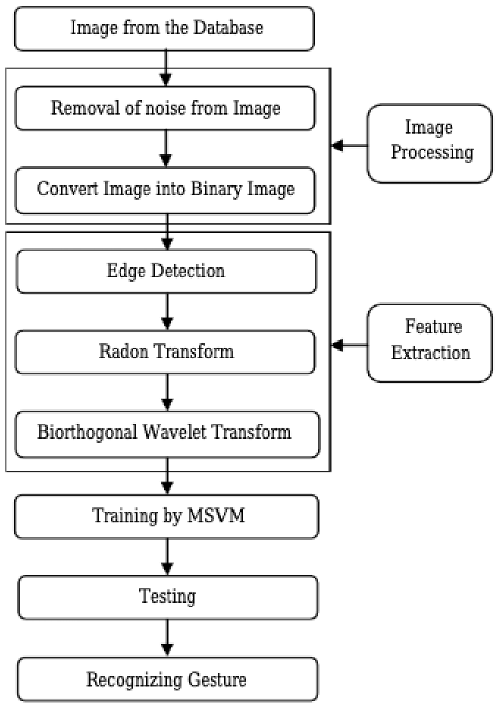


Figure 5. Graphical representation illustrating the components and functions of a system designed to identify and interpret hand gestures.

The Canny Edge Detection Algorithm is used in this system to detect the edges of an image because it outperforms other reference algorithms. The algorithm aims to determine the most suitable edge by minimizing the error rate, maximizing edge location accuracy, and marking edges only one time if there is one edge for the minimum response. The optimal filter that satisfies all of these criteria can be approximated by using the first derivative of a Gaussian function.

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{\frac{x^2+y^2}{2\sigma^2}}, \tag{8}$$

$$\frac{\partial G(x,y)}{\partial x} \alpha \, x e^{-\frac{x^2+y^2}{2\sigma^2}}, \tag{9}$$

$$\frac{\partial G(x,y)}{\partial x} \alpha\, y e^{-\frac{x^2+y^2}{2\sigma^2}}, \tag{10}$$

During the first stage of the process, the image is passed through a Gaussian filter using convolution. This produces a convoluted image which is used to measure the gradient of the original image. The 2-dimensional convolution operation is defined by the following equation.

$$I'(x,y) = g(k,l) \times I(x,y), \tag{11}$$

$$l'(x,y) = \sum_K^N l = -N \sum_l^N l = -N^{g(k,l)I(x-k,y-l)}, \tag{12}$$

where:
$g(k,l)$ = convolutional kernel
$I(x,y)$ = original image
$I'(x,y)$ = filtered image
$2N+1$ = size of convolutional kernel

The Canny Edge Detection process involves several steps, as shown in Figure 6. First, the input image's noise is removed, and then the image is converted into a binary image. Then, the Canny Edge Detection algorithm is employed to identify the boundary of the hand.

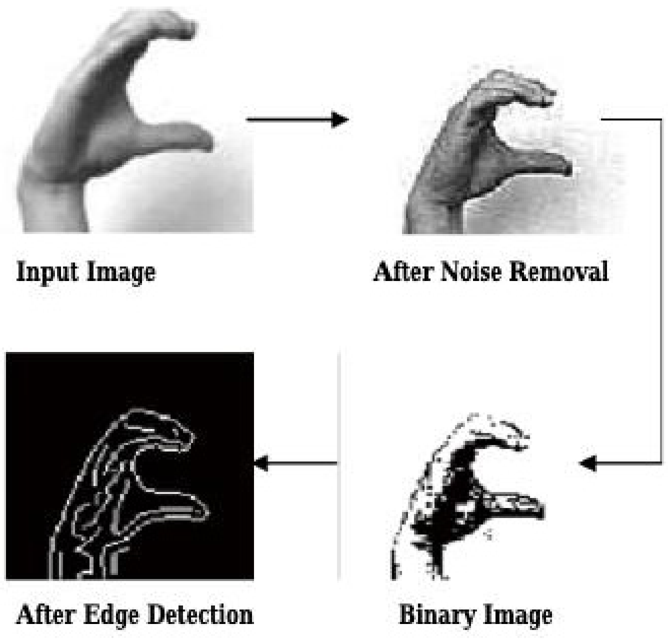


Figure 6. The process of detecting edges using the Canny Edge Detection algorithm from an input image.

The proposed method was tested on ten different hand gestures performed by three users, as shown in Figure 4. Table 2 presents the recognition results for three of these users across the ten gestures. The accuracy vs. user curve is depicted in Figure 5 for all five users. The results indicate that the suggested algorithm is effective in precisely identifying hand gestures. To ensure the method's robustness against varying operating conditions such as illumination, posture, zooming, and skin color, a large dataset of hand gestures was employed to train the classifier. The training set used in the detection phase consisted of over 978 positive samples and 688 negative image samples.

DOI: 10.37943/15LPCU4095
© Samat Mukhanov, Raissa Uskenbayeva, Young Im Cho, Kabyl Dauren, Les Nurzhan, Maqsat Amangeldi

In [12] the paper authors explores contour analysis in computer vision, highlighting the significance of active contour methods, particularly the minimum energy curve. The Canny Boundary Detector algorithm is applied to detect object contours by reducing image blur and eliminating noise. Machine learning, specifically clustering, plays a vital role in recognition and border identification. Mathematical models unique to clustering are employed for contour recognition. Additionally, the paper utilizes graph analysis in contour detection and linking, maintaining efficiency in noisy environments. The convexity defects contour analysis algorithm measures border sizes and identifies contour recesses while calculating object parameters.

Models of architecture neural network presents a machine learning method in a deep neural network. EffectiveNet is a composite scaling method, which consistently improves model accuracy and efficiency for scaling existing models such as MobileNet (+1.4 % image fidelity) and ResNet (+0.7 %) over traditional scaling methods [13,14].

Results and Discussion

Applying and analyzing research experience from scientific publications and experiments, the focus was on testing the hypotheses proposed by authors of various high-impact journal articles. Engineering approach in exploring machine learning and neural network methods, gathering and preparing computer vision data, and using the appropriate libraries for machine learning and deep learning tasks, such as SVM, LSTM, and CNN models have been applied. All the work was conducted within the Jupyter Notebook environment, which provides a powerful tool for data manipulation, coding, training, and exploration in an interactive and multimedia environment, making it a popular tool in the fields of science, research, and education. Additionally, relevant metrics, such as the confusion matrix, were applied during the training of recurrent and convolutional neural network models.

The method proposed in this study was tested on ten different hand gestures (A, B, C, D, G, H, I, N, O, P) performed by three users, as illustrated in Figure 7. The recognition results for three of these users across the ten gestures are presented in Table 1. Figure 7 shows the accuracy vs. user curve for all five users. These results indicate that the proposed algorithm is capable of accurately detecting hand gestures. To ensure that the method is robust to varying operating conditions, a large dataset of hand gestures was utilized to train the classifier. The training set used in the detection phase included more than 978 positive samples and 688 negative image samples.

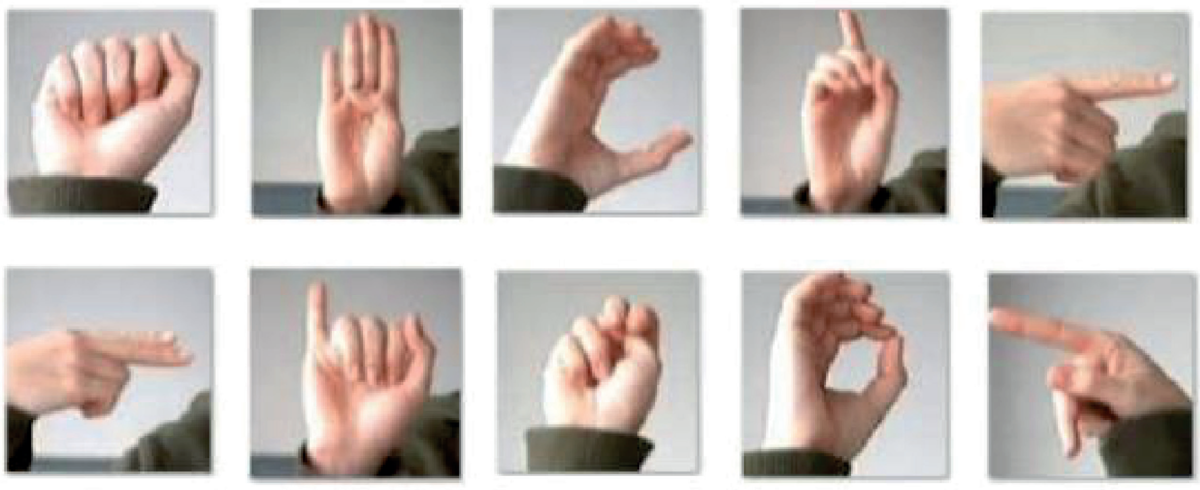


Figure 7. Finger spelled Alphabet (Top row [A, B, C, D, G], Bottom row [H, I, N, O, P]).

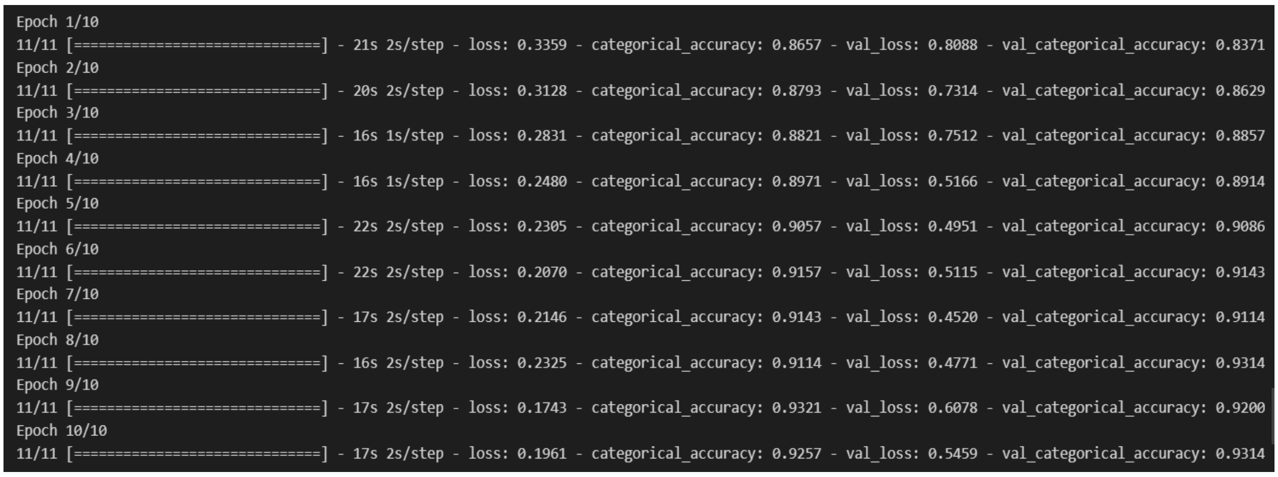
Scientific Journal of Astana IT University
ISSN (P): 2707-9031 ISSN (E): 2707-904X
VOLUME 15, SEPTEMBER 2023

The table below demonstrates accuracy of effectives of gesture hand recognition.

Table 1. Recognition results of 10 gestures of 3 users.
In every letter there is percentage of accuracy hand and sign language recognition.

User	A	B	C	D	G	H	I	N	O	P
1	88%	83%	88%	68%	68%	73%	77%	75%	88%	75%
2	86 %	78%	85%	70%	75%	61%	74%	79%	82%	82%
3	87 %	80%	82%	63%	63%	70%	77%	71%	93%	87%
Total	87%	80%	85%	67%	68%	68%	76%	75%	87%	81%

During the training process, the model achieved an impressive accuracy of 92% on the testing set, indicating that it can correctly identify the gesture class for most of the images. This high accuracy is a promising result and suggests that the model has effectively learned the patterns and features necessary to classify hand gestures in Kazakh sign language.

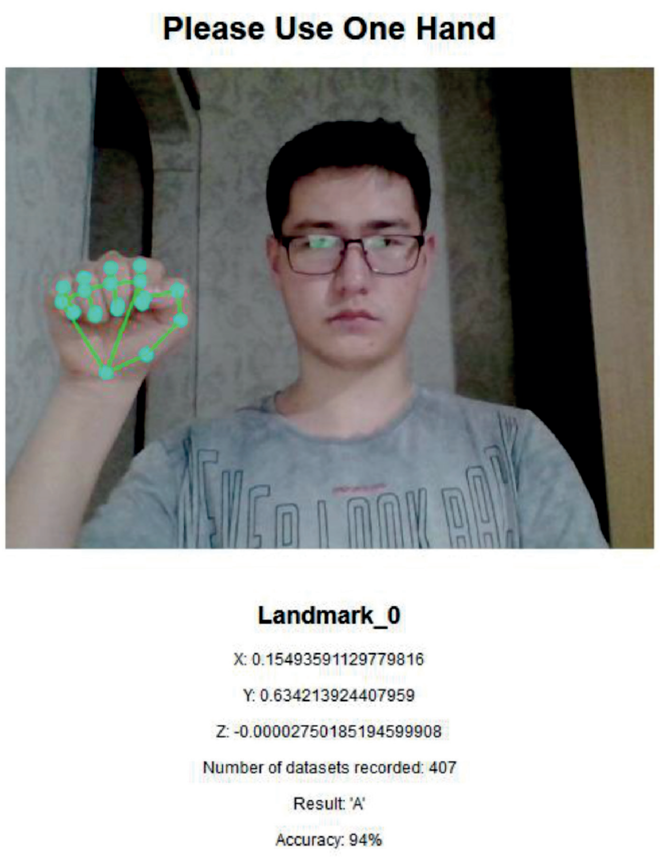


```
Epoch 1/10
11/11 [==============================] - 21s 2s/step - loss: 0.3359 - categorical_accuracy: 0.8657 - val_loss: 0.8088 - val_categorical_accuracy: 0.8371
Epoch 2/10
11/11 [==============================] - 20s 2s/step - loss: 0.3128 - categorical_accuracy: 0.8793 - val_loss: 0.7314 - val_categorical_accuracy: 0.8629
Epoch 3/10
11/11 [==============================] - 16s 1s/step - loss: 0.2831 - categorical_accuracy: 0.8821 - val_loss: 0.7512 - val_categorical_accuracy: 0.8857
Epoch 4/10
11/11 [==============================] - 16s 1s/step - loss: 0.2480 - categorical_accuracy: 0.8971 - val_loss: 0.5166 - val_categorical_accuracy: 0.8914
Epoch 5/10
11/11 [==============================] - 22s 2s/step - loss: 0.2305 - categorical_accuracy: 0.9057 - val_loss: 0.4951 - val_categorical_accuracy: 0.9086
Epoch 6/10
11/11 [==============================] - 22s 2s/step - loss: 0.2070 - categorical_accuracy: 0.9157 - val_loss: 0.5115 - val_categorical_accuracy: 0.9143
Epoch 7/10
11/11 [==============================] - 17s 2s/step - loss: 0.2146 - categorical_accuracy: 0.9143 - val_loss: 0.4520 - val_categorical_accuracy: 0.9114
Epoch 8/10
11/11 [==============================] - 16s 2s/step - loss: 0.2325 - categorical_accuracy: 0.9114 - val_loss: 0.4771 - val_categorical_accuracy: 0.9314
Epoch 9/10
11/11 [==============================] - 17s 2s/step - loss: 0.1743 - categorical_accuracy: 0.9321 - val_loss: 0.6078 - val_categorical_accuracy: 0.9200
Epoch 10/10
11/11 [==============================] - 17s 2s/step - loss: 0.1961 - categorical_accuracy: 0.9257 - val_loss: 0.5459 - val_categorical_accuracy: 0.9314
```

Figure 8. Compiling model results

Evaluation of the trained model was carried out using the confusion matrix, which provides a visual representation of the model's performance. The confusion matrix revealed that the model performed well for most of the classes, with some classes having lower accuracy than others. This analysis can help identify the classes that need more attention during the training process to improve the overall accuracy of the model. Overall, the training and evaluation of the CNN model for hand gesture recognition was successful, with a high accuracy rate achieved on the testing set. The trained model can now be integrated into a real-world application for hand gesture recognition, bringing the benefits of technology to people with hearing impairments who use Kazakh sign language. Real-time testing is a crucial step in ensuring the accuracy and reliability of our hand gesture recognition system as part of our diploma project. To perform real-time testing, we integrated the trained CNN model with the React front-end, which enables users to make hand gestures in front of the camera. TensorFlow.js for hand pose tracking and segmentation was used, which was a challenging task due to the variability in lighting conditions, camera angles, and backgrounds. Below figures demonstrates testing methods of hand gesture recognition for Kazakh sign language:

DOI: 10.37943/15LPCU4095



Please Use One Hand

Landmark_0

X: 0.15493591129779816

Y: 0.634213924407959

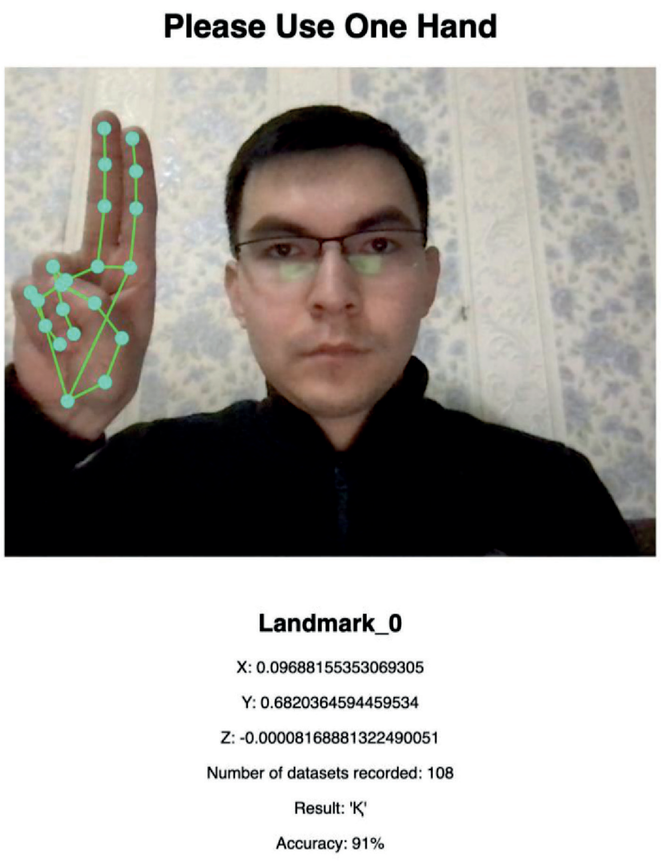
Z: -0.000027501851945999908

Number of datasets recorded: 407

Result: 'A'

Accuracy: 94%

Figure 9. Result of testing. Gesture "A"



Please Use One Hand

Landmark_0

X: 0.09688155353069305

Y: 0.6820364594459534

Z: -0.00008168881322490051

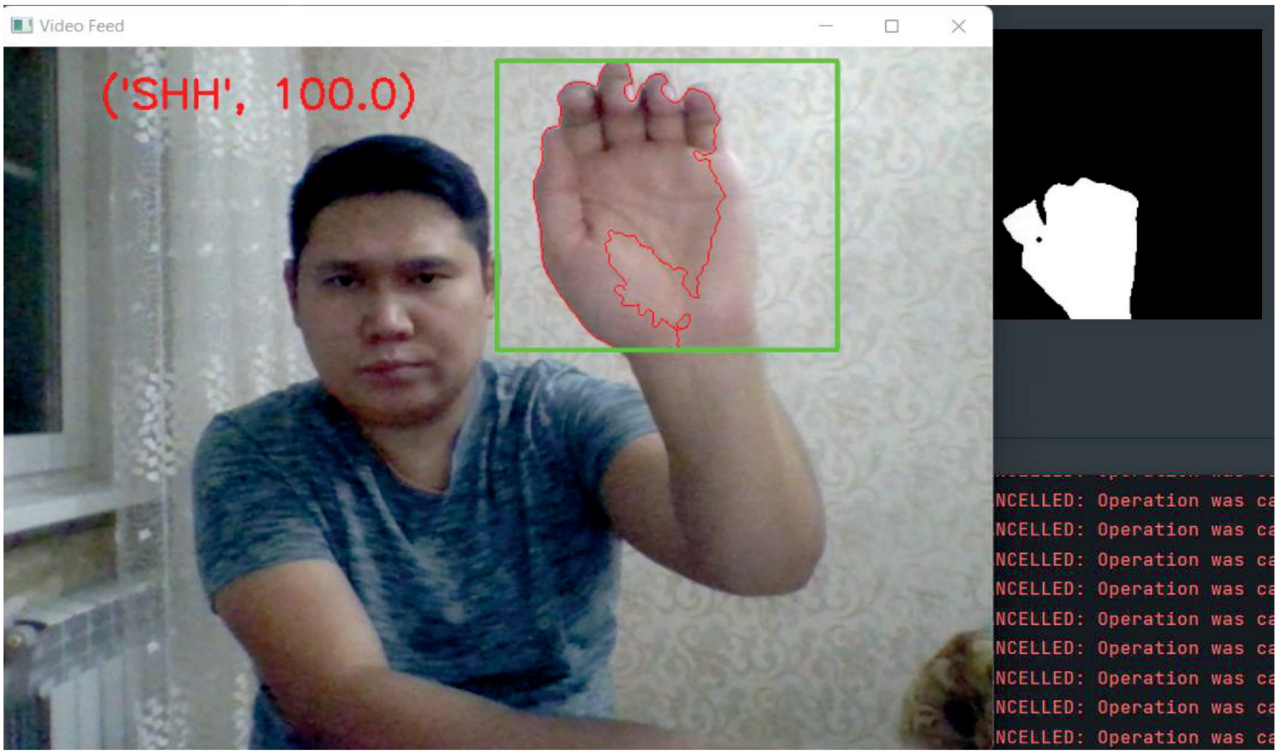
Number of datasets recorded: 108

Result: 'Қ'

Accuracy: 91%

Figure 10. Result of testing. Gesture "Қ"

The picture below represents switching on camera and tracking objects in real time and implements CNN method - high quality gesture recognition. Firstly tested training model of CNN completed without bugs and errors and neural network successfully trained with high trained accuracy and valid values. After, it was integrated in Pyrthon PyCharm IDE and launched it to look the results of effective working gesture recognizer:



```
NCELLED: Operation was ca
NCELLED: Operation was ca
NCELLED: Operation was ca
NCELLED: Operation was ca
NCELLED: Operation was ca
NCELLED: Operation was ca
NCELLED: Operation was ca
NCELLED: Operation was ca
NCELLED: Operation was ca
```

Figure 11. Result of testing with threshold and gray scale. Gesture "Щ"

During testing, it became clear that the system achieved high accuracy in recognizing hand gestures in Kazakh sign language, with an average accuracy of 89%. However, several challenges in real-world scenarios that affected the system's performance were also identified. For example, the system struggled to recognize hand gestures accurately in low-light conditions, and it was sensitive to camera angles and backgrounds. In order to address these challenges, the hand pose tracking and segmentation algorithms were optimized, and the CNN model's hyperparameters to improve the model's robustness were adjusted. Furthermore, there was an experiment with different image preprocessing techniques and explored different CNN architectures to improve the system's performance. Overall, the real-time testing results were promising, indicating that the system has the potential to be used in real-world applications for hand gesture recognition in Kazakh sign language. However, further testing and optimization are necessary to ensure the system's reliability and accuracy in various environments and situations. In conclusion, the development of a hand gesture recognition system using a convolutional neural network involves several critical steps, including data preparation, model training, and integration with the front-end application. Collecting, normalizing, and labeling datasets, along with careful segmentation, conversion to grayscale, and resizing of the images, is crucial for effective model training. The use of modern tools like TensorFlow.js and React allows for seamless integration of hand pose tracking and segmentation, as well as asynchronous requests with the trained model. The real-time testing of the system shows promising results with high accuracy in recognizing hand gestures in Kazakh sign language. Overall, the successful development of a hand gesture recognition system can have significant applications in improving communication accessibility for people with hearing or speech impairments.

Conclusion

Concluding this research work, result of training methods and models of recurrent and neural network for gesture recognition of Kazakh Sign Language have been reached. In this work machine learning methods and pattern recognition algorithms, specifically for gestures, were explored. Among them, LSTM demonstrated the best and most effective performance, followed by CNN, and then SVM. Experiments were conducted under various visibility and distance conditions from the webcam. Different gestures yielded different recognition results, as each gesture is unique, and their representation varies from simple to complex hand or palm positions. This research also presents both the results and tests during the training and recognition of gestures in the Kazakh sign language. Various recognition results are separately documented in tables. This paper provides an overview and comparison of various feature neural networks including CNN, LSTM, and SVM for hand gesture recognition. The performance of three popular neural network models (LSTM, CNN, and SVM) was compared, and the following conclusions were made: LSTM, which is a type of recurrent neural network (RNN), has shown promising results in sequence modeling tasks. In hand gesture recognition, LSTM can capture the temporal information of hand movements, making it useful for recognizing complex gestures that involve multiple movements. However, LSTM may require a substantial amount of training data to achieve optimal performance. On the other hand, convolutional neural networks (CNNs) are commonly used in image recognition tasks and can be applied to hand gesture recognition. CNNs can automatically learn features from raw input data, including the shape and position of the hand. Furthermore, CNNs can handle data with different resolutions, making them suitable for recognizing hand gestures in various contexts. However, CNNs might not be as effective as LSTM in capturing the temporal information of hand movements.

References

[1] Vidyanova, A. (2022). In the USA, they are interested in the development of Kazakhs for the deaf. *Capital.* https://kapital.kz/business/105455/v-ssha-zainteresovalis-razrabotkoykazakh-stantsev-dlya-glukhikh.html.

[2] Bazarevsky, V., & Fan, Zh. (2019). On-device, real-time hand tracking with mediapipe. *Google AI Blog*. https://ai.googleblog.com/2019/08/on-device-real-time-hand-tracking-with.html.

[3] Wang, Y., Wang, H., & He, X. (2020). Sign language recognition based on deep convolutional neural network. *IEEE Access*, 8, 64990-64999. https://doi.org/10.3390/electronics12040786

[4] Lee, A.R., Cho, Y., Jin, S., & Kim, N. (2020). Enhancement of surgical hand gesture recognition using a capsule network for a contactless interface in the operating room. *Computer methods and programs in biomedicine,* 190, 105385. https://doi.org/10.1016/j.cmpb.2020.105385

[5] Bilgin, M., & Mutludogan, K. (2019). American Sign Language character recognition with capsule networks. *International Symposium on Multidisciplinary Studies and Innovative Technologies*, 3. https://doi.org/10.1109/ismsit.2019.8932829

[6] Kudubayeva, S.A., Ryumin, D.A., & Kalzhanov, M.U. (2016). The method of basis vectors for recognition sign language by using sensor KINECT. *Journal of Mathematics, Mechanics and Computer Science*, 91(3). https://bm.kaznu.kz/index.php/kaznu/article/view/541

[7] Adithya, V., & Reghunadhan, R. (2020). A deep convolutional neural network approach for static hand gesture recognition. *Procedia Computer Science*, 171, 2353-2361. https://doi.org/10.1016/j.procs.2020.04.255.

[8] Lai, K., & Yanushkevich, S.N. (2018). CNN+RNN depth and skeleton based dynamic hand gesture recognition. *International Conference on Pattern Recognition (ICPR), IEEE,* 24. https://doi.org/10.1109/ICPR.2018.8545718

[9] Merembayev, T., Kurmangaliyev, D., Bekbauov, B., & Amanbek, Y. (2021). A Comparison of Machine Learning Algorithms in Predicting Lithofacies: Case Studies from Norway and Kazakhstan. *Energies*, 14(7), 1896. https://doi.org/10.3390/en14071896

[10] Mantecón, T., del Blanco, C.R., Jaureguizar, F., & García, N. (2016) Hand gesture recognition using infrared imagery provided by leap motion controller. *Int. Conf. on Advanced Concepts for Intelligent Vision Systems*, 47-57, 24-27. https://doi.org/10.1007/978-3-319-48680-2_5

[11] Kumar, A., Thankachan, K., & Dominic, M.M. (2016) Sign language recognition. *IEEE international conference on recent advances in information technology (RAIT),* 3, 422–428. https://doi.org/10.1109/rait.2016.7507939.

[12] Uskenbayeva, R.K., & Mukhanov, S.B. (2020). Contour analysis of external images. International Conference on Engineering & MIS, 6, 1–6. https://doi.org/10.1145/3410352.3410811

[13] Tan, M., & Le, Q. (2019). Efficientnet: Rethinking model scaling for convolutional neural networks. *International Conference on Machine Learning*, 6105-6114. https://arxiv.org/abs/1905.11946

[14] Kenshimov, C., Mukhanov, S., Merembayev, T., & Yedilkhan, D. (2021). A comparison of convolutional neural networks for Kazakh sign language recognition. *Eastern-European Journal of Enterprise-Technologies*, 5 (2(113)), 44–54. https://journals.uran.ua/eejet/article/view/241535