

DOI: 10.37943/18JYLU4904**Samat Mukhanov***

PhD, Senior-lecturer, Department of Computer Engineering
s.mukhanov@iitu.edu.kz, orcid.org/0000-0001-8761-4272
International Information Technology University, Kazakhstan

Raissa Uskenbayeva

Doctor of technical science, Professor, Vice-Rector for Academic Affairs
r.k.uskenbayeva@satbayev.university, orcid.org/0000-0002-8499-2101
Satbayev University, Kazakhstan

Abdul Ahmad Rakhim

PhD, Professor, Department of Computing and Informatics
abdrahim@uniten.edu.my, orcid.org/0000-0001-7923-0105
Universiti Tenaga Nasional, Malaysia

Young Im Cho

PhD, Professor, Faculty of Computer Engineering
yicho@gachon.ac.kr, orcid.org/0000-0003-0184-7599
Gachon University, Korea

Aknur Yemberdiyeva

Master of Technical Sciences, Lecturer, Department of Computer
Engineering
a.yemberdiyeva@iitu.edu.kz, orcid.org/0009-0005-5078-2412
International Information Technology University, Kazakhstan

Zhansaya Bekaulova

Master of Technical Sciences, Senior-lecturer, Department of Computer
Engineering
zh.bekaulova@iitu.edu.kz, orcid.org/0009-0000-9339-9222
International Information Technology University, Kazakhstan

DEEP AND MACHINE LEARNING MODELS FOR RECOGNIZING STATIC AND DYNAMIC GESTURES OF THE KAZAKH ALPHABET

Abstract: Currently, an increasing amount of research is directed towards solving tasks using computer vision libraries and artificial intelligence tools. Most common are the solutions and approaches utilizing machine and deep learning models of artificial neural networks for recognizing gestures of the Kazakh sign language based on supervised learning methods and deep learning for processing sequential data. The research object is the Kazakh sign language alphabet aimed at facilitating communication for individuals with limited abilities. The research subject comprises machine learning methods and models of artificial neural networks and deep learning for gesture classification and recognition. The research areas encompass Machine Learning, Deep Learning, Neural Networks, and Computer Vision.

The main challenge lies in recognizing dynamic hand gestures. In the Kazakh sign language alphabet, there are 42 letters, with 12 of them being dynamic. Processing, capturing, and recognizing gestures in motion, particularly in dynamics, pose a highly complex task. It is imperative to employ modern technologies and unconventional approaches by combining various recognition methods/algorithms to develop and construct a hybrid neural network model for gesture recognition. Gesture recognition is a classification task, which is one of the directions

of pattern recognition. The fundamental basis of recognition is the theory of pattern recognition. The paper discusses pattern recognition systems, the environment and application areas of these systems, and the requirements for their development and improvement. It presents tasks such as license plate recognition, facial recognition, and gesture recognition. The field of computer vision in image recognition, specifically hand gestures, is also addressed. The development of software will enable the testing of the trained model's effectiveness and its application for laboratory purposes, allowing for adjustments to improve the model.

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

Introduction

Pattern recognition theory is a field of study within artificial intelligence and computer vision that deals with the development of methods and algorithms for automatic recognition and classification of objects and patterns based on their characteristics and features. This field has a broad spectrum of applications, including text recognition, image recognition, sound recognition, voice recognition, face recognition, gesture recognition, and many other types of data. In the domestic literature, education, and science in the field of pattern recognition, the author of the book "Pattern Recognition Theory and Cluster Analysis," E.N. Amirgaliev, presented in mathematical form problems of classification, cluster analysis, pattern recognition algorithms, and machine learning methods [1]. This book describes fundamental concepts, theorems, definitions, as well as mathematical formulas of algorithmic equations. A foreign counterpart to this work is Christopher M. Bishop's book on pattern recognition and machine learning.

The process of pattern recognition is a cognitive mechanism studied in the fields of psychology and cognitive neurobiology. This process involves matching the input information received from the stimulus with the data stored in memory [2]. In pattern recognition systems, the physical characteristics of patterns can be either simple or complex. An example of a simple pattern recognition system is the classification of pixels in multispectral scanners or digitized images, where the features are the spectral reflection properties of objects on the Earth's surface. However, when using other types of data in the classification process, the system becomes more complex. Complex pattern recognition systems can be single-layered or multi-layered.

Improving recognition accuracy requires the development of new algorithms and machine learning methods capable of providing higher accuracy in the recognition process [3]. Additionally, focusing on recognizing more complex gestures involves creating systems capable of identifying not only simple gestures but also more complex variations, such as gestures composed of multiple components or combinations of several simple gestures [4].

Kazakh Sign Language Alphabet

The Kazakh dactyl (sign) alphabet is a system of sign communication used by deaf and hard of hearing individuals in the Kazakh language. In this system, 42 signs represent letters of the Kazakh alphabet and punctuation marks. Each sign is formed using combinations of finger, hand, and wrist movements [5]. For example, to denote the letter "A," a sign with spread fingers and raised hands is used, while for the letter "B," a circular motion of the clenched right hand is performed. Various variants of the Kazakh dactyl alphabet are used in different regions inhabited by Kazakhs [6]. Training in this system is conducted in specialized schools and centers for the deaf and hard of hearing. In Kazakhstan, approximately 200,000 people have hearing impairments [7]. According to the legislation of the Republic of Kazakhstan, each person with a hearing impairment is provided with free sign language specialist services for 60 hours per year, although this may be insufficient [8]. The importance of this topic is driven by the com-

mon interest of the state, sign language representatives, and parents of children with hearing impairments. Research in this area has a social focus, for example, helping people with disabilities [9].

The present study focuses on creating a system for translating regular language into sign language, with the ability to convey the semantic meaning of the translation from the Kazakh language to Kazakh Sign Language (KSL). Below is the Kazakh Sign Language Alphabet presented in image format for each letter:











































 Аа	 Әә	 Бб	 Вв	 Гг	 Ғғ	 Дд	 Ее
 Ёё	 Жж	 Зз	 Ии	 Йй	 Кк	 Ққ	 Лл
 Мм	 Нн	 Ңң	 Оо	 Өө	 Пп	 Рр	 Сс
 Тт	 Уу	 Үү	 Үү	 Фф	 Хх	 Һһ	 Цц
 Чч	 Шш	 Щщ	 Ыы	 Іі	 Ъъ	 Ыы	 Ээ
 Юю	 Яя						

Figure 1. Kazakh Dactyl (Sign) Alphabet

Figure 1 presents the Kazakh sign alphabet in the form of images for each letter [10][11]. In total, the alphabet consists of 42 letters, meaning there are 42 classes for gesture classification tasks. This alphabet has been used for training deep neural network models [12][13][14]. It comprises letters represented as images and serves as an excellent opportunity for data collection and preparation (datasets) for training neural networks.

Model and methods

Hand gestures can be described in terms of Euclidean space, where each gesture is represented by a vector in this space [15][16][17]. Let's consider a more detailed mathematical description of hand gestures using formulas and equations. Assume we have n key points on the hand, and each point is represented by a three-dimensional coordinate (x, y, z) in three-dimensional geometric space.

Defining Key Point Vectors:

For each point, we define the key point vector $P_i = (x_i, y_i, z_i)$, where i is the index of the point.

Defining the Gesture Feature Vector:

The gesture feature vector F will be represented by a vector that includes the coordinates of all the key points of the hand:

$$F = (p_1, p_2, \dots, p_n), \quad (1)$$

Gesture Space:

The gesture space will represent a linear subspace of three-dimensional geometric space, defined by the gesture feature vectors F .

Gesture Classification:

Each gesture corresponds to a specific class or category, which is determined based on the analysis of the feature vector F [18].

Model Training:

To train a gesture classification model, we use a training dataset where each gesture is represented by a feature vector F and its corresponding class. The model training can include machine learning methods such as SVM, CNN, LSTM, and others [19].

Gesture Recognition:

After training the model, we can use it to recognize hand gestures by inputting the feature vector F . The model will classify the gesture, determining the corresponding class. Thus, linear algebra in geometric space allows abstracting and describing various hand gestures in a mathematical representation using vectors and linear operations. This enables training models for recognizing and classifying gestures.

Calculating Distances Between Key Points:

For gesture analysis, we can calculate the distances between key points. The distance between two points P_i and P_j is calculated using the Euclidean distance formula:

$$d(P_i, P_j) = (x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2, \quad (2)$$

This distance can be used as one of the features for gesture analysis.

Angles Between Hand Joints:

To evaluate the angles between hand joints, trigonometry can be used. For example, the angle

θ between the vectors $P_i - P_j$ and $P_j - P_k$ can be calculated using the dot product and the norms of the vectors:

$$\cos(\theta) = \frac{|P_i - P_j| \cdot |P_j - P_k|}{|(P_i - P_j) \cdot (P_j - P_k)|}, \quad (3)$$

where θ is the angle between the vectors.

Gesture Recognition in the Geometric Plane:

For simplifying gesture analysis, the coordinates of the key points of the hand can be projected onto a two-dimensional geometric plane. This allows working with two-dimensional vectors and simplifies calculations.

Dataset, data augmentation

The project aimed to develop a reliable hand gesture recognition system using CNNs. To achieve this, a dataset of 42 Kazakh Sign Language classes was collected, followed by segmentation, which involved isolating the hand gesture region from the background using a thresholding algorithm [20]. The resulting images were then converted to grayscale, which simplifies CNN processing and improves contrast. Resizing the images to a standard size was crucial for effective comparison during training [21]. Labeling the dataset with correct class information was essential for the CNN to associate specific hand gestures with their corresponding labels [22]. This process, which can be time-consuming, is crucial for developing an accurate and reliable hand gesture recognition system. Low-quality images were removed from the dataset, resulting in only high-quality images.

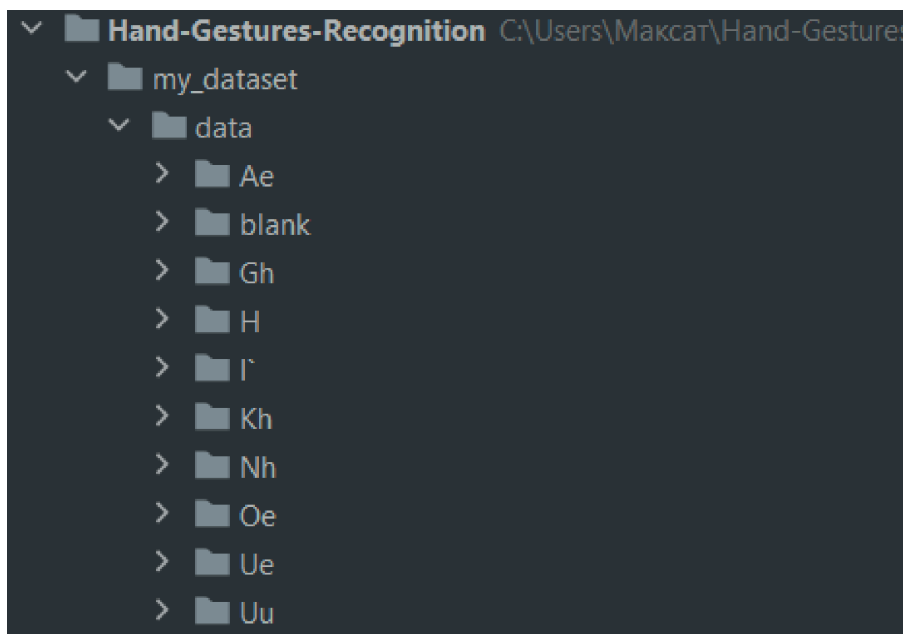


Figure 2. Dataset collection

As we apply data augmentation using the ImageDataGenerator, it generates augmented versions of our original dataset on-the-fly during training. These augmented images are used to train our model, effectively increasing the diversity of the training dataset [23]. This helps our model generalize better to variations in the input data, making it more robust when faced with unseen examples.

In our case, the ImageDataGenerator is configured with various augmentation techniques such as rotation, width and height shifts, shear, zoom, and horizontal flip [24]. As our model trains, it doesn't see the same image multiple times but encounters different augmented versions of the original images in each epoch.

So, the dataset is not physically changed; rather, during each training epoch, the model is exposed to different views of the original images, which simulates a larger and more diverse dataset [25]. This process helps improve this model's ability to handle variations and perform well on unseen data.

The Experimental results of machine and deep learning models

Support Vector Machines

Multiclass Support Vector Machines (SVMs) offer a powerful and versatile approach to hand gesture recognition with multiple classes. Their ability to handle diverse sets of gestures and leverage the advantages of binary SVMs makes them valuable for various applications. Multiclass SVMs extend the capabilities of binary SVM classification for simultaneous recognition of multiple hand gestures [26].

A trained SVM classifies an unseen gesture based on its feature representation and hyperplanes. In the case of One-vs-One or Directed Acyclic Graph (DAG) strategies, various voting schemes can be employed to combine results from multiple SVMs. Several advantages of binary SVM classification for hand gesture recognition are noted, such as direct classification of multiple gestures, eliminating the need for sequential binary classification [27].

The flexibility in handling large sets of gestures allows adaptation to diverse hand gestures and complex sign languages. Multiclass SVMs leverage the advantages of existing binary SVMs, maintaining high accuracy, robustness, and good performance with large datasets.

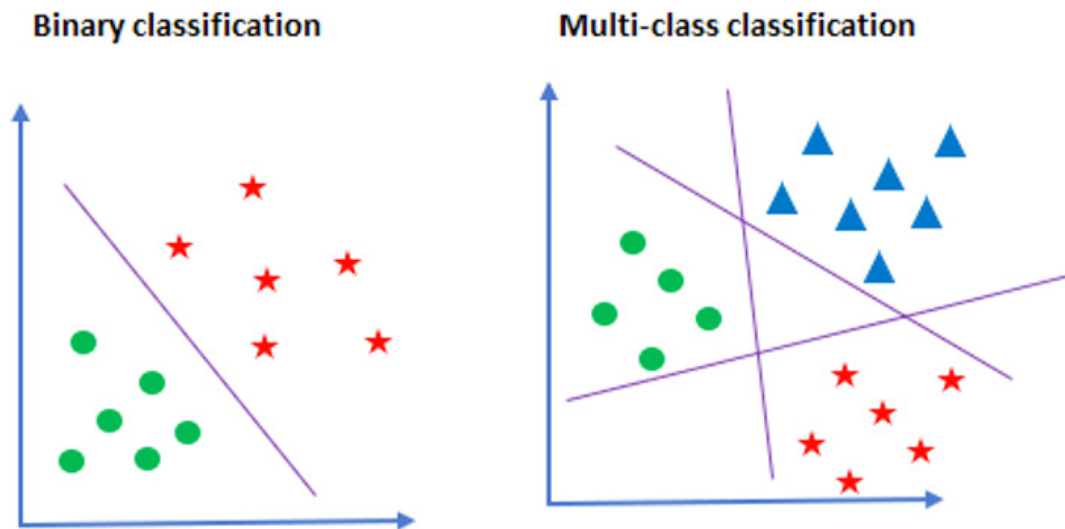


Figure 3. Comparing of SVM Binary and Multi-class classification

Performance Evaluation and Comparison.

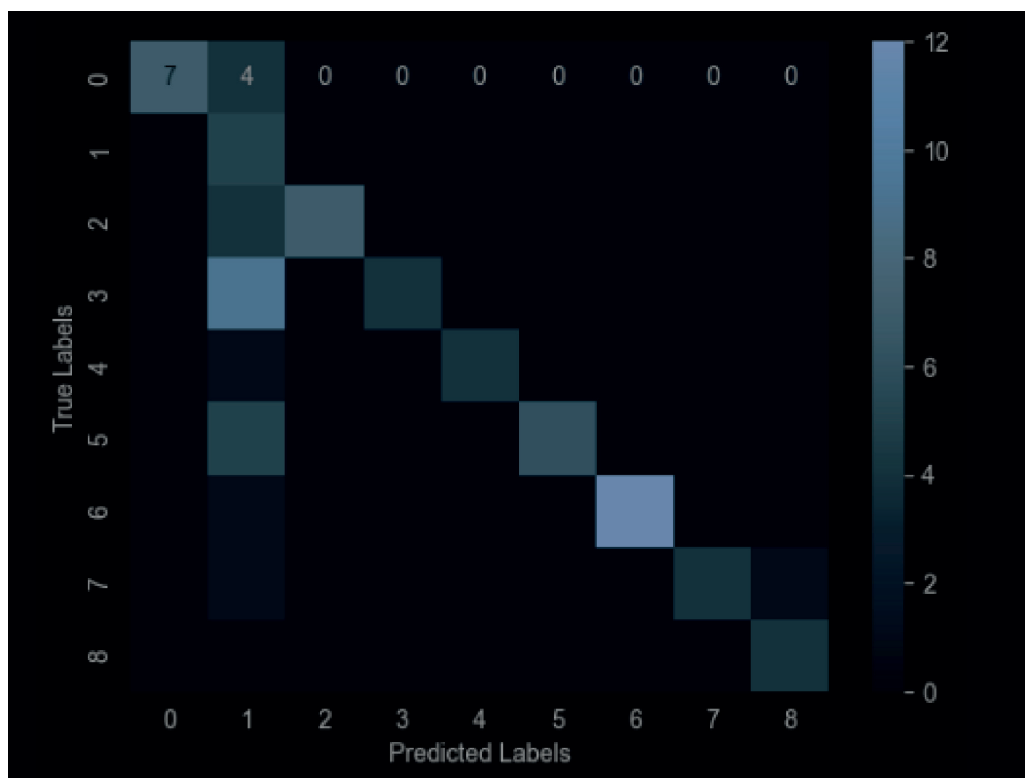


Figure 4. Confusion Matrix for the 9 classes of the Kazakh Sign Alphabet

The confusion matrix reveals the model's performance in terms of misclassification. It shows that the model correctly predicts Class 0 for all cases, indicating perfect classification accuracy for this class. However, it struggles with Class 1, misclassifying 4 cases as Class 0. The model performs well in correctly predicting all classes without errors, except for Class 1.

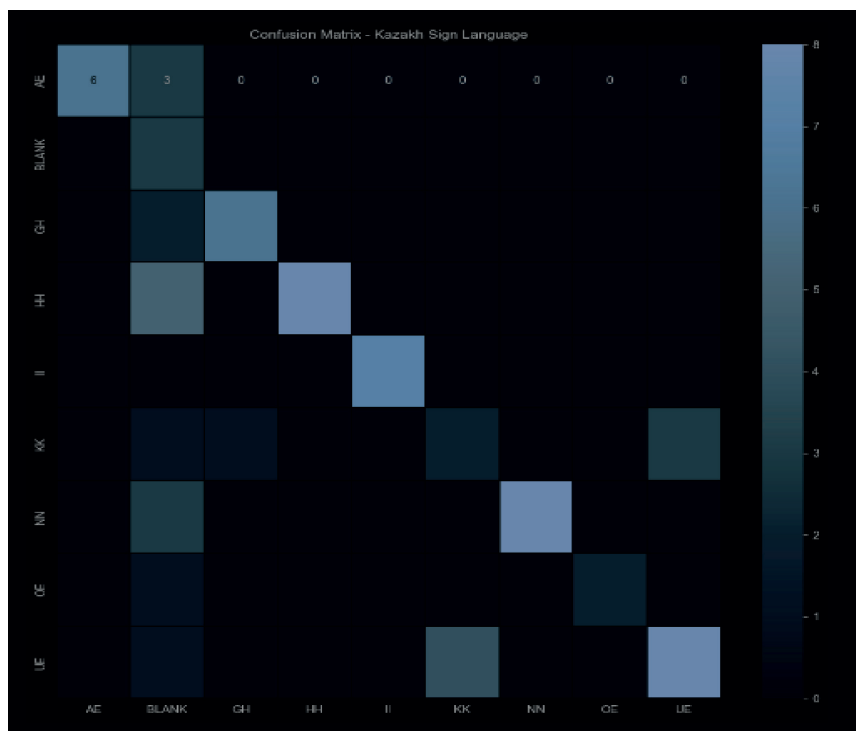


Figure 5. The second Confusion Matrix of Kazakh Letters in the Kazakh Sign Alphabet

The second confusion matrix provides another perspective on the model's performance. It shows that the model correctly predicts all cases. The model performs well in predicting all classes, with only one error in each class except for Class 1.

Table 1. Metrics for Evaluating Binary SVM Model Classification

Class	Precision	Recall	F1 score	Support
1	2	3	4	5
Ae(Ə)	0.69	0.64	0.78	11
Blank	0.71	0.91	0.29	5
Gh(Ғ)	0.71	0.64	0.78	11
Hh(Һ)	0.65	0.31	0.47	13
Ii(І)	0.81	0.80	0.89	5
Kk(Қ)	0.77	0.55	0.71	11
Nn(Ң)	0.79	0.81	0.91	13
Oe(Ө)	0.72	0.67	0.80	6
Ue(Ү)	0.72	0.80	0.89	4
Accuracy	0.74			79
Macro Avg	0.89	0.72	0.73	79
Weighted Avg	0.94	0.67	0.73	79

The multiclass model demonstrated clear progress in accuracy, recall, and overall performance compared to the previous version. The enhanced model is more reliable in positive predictions, has a better ability to capture positive cases, and maintains a balanced approach between precision and recall. While the initial model laid the foundation, the recent improvements demonstrate the effectiveness of ongoing efforts to optimize and tune the character recognition system for various classes.

Table 2. Metrics for Evaluating Multiclass SVM Model Classification

Class	Precision	Recall	F1 score	Support
Ae(Ә)	0.89	0.62	0.78	11
Blank	0.89	1.00	0.29	5
Gh(Ғ)	0.79	0.64	0.78	11
Hh(Һ)	0.72	0.31	0.47	13
Ii(І)	0.82	0.76	0.89	5
Kk(Қ)	0.81	0.55	0.71	11
Nn(Ң)	0.69	0.92	0.96	13
Oe(Ө)	0.81	0.67	0.80	6
Ue(Ү)	0.82	1.00	0.89	4
Accuracy	0.81			79
Macro Avg	0.90	0.72	0.73	79
Weighted Avg	0.95	0.67	0.73	79

Continued refinement and attention to class-specific characteristics are likely to lead to further development in the future.

The Long Short-Term Memory model

This method was investigated and trained on collected images of the Kazakh sign alphabet using neural networks with long short-term memory (LSTM), TensorFlow, and MediaPipe (a skeletonization method). The study examines the effective integration of these technologies for interpreting both static and dynamic gestures within the Kazakh sign alphabet [28].

LSTM uses sequences of data for recognition. Since we are applying recognition to video, our sequence consists of video frames of hands [29]. In each video frame, the presence of a hand and its position need to be determined [30]. Once this data is obtained, we can train our network using the LSTM model. To solve the problem of determining hand positions in frames, we employed MediaPipe technology developed by Google.

MediaPipe is an open-source platform developed by Google that provides a pipeline of machine learning models for processing audio, video, and sensor data [31]. It offers a collection of machine learning models for tasks such as object detection, face detection, pose estimation, hand tracking, and more [32]. MediaPipe provides numerous building blocks for constructing machine learning pipelines capable of processing multimedia data in real time, including pre-processing, feature extraction, model inference, and postprocessing [33].

According to Figure 6, MediaPipe can locate the position of a hand in an image and create its skeleton model based on 21 joints and their connections [34]. Each joint in the image has its coordinate point (x, y, z) , where x and y represent the distance from the bottom-left corner of the image (the origin), and z represents the depth of the joint. This results in 63 values for the 21 hand joints. Since we are using both hands, each frame will have 126 inputs.

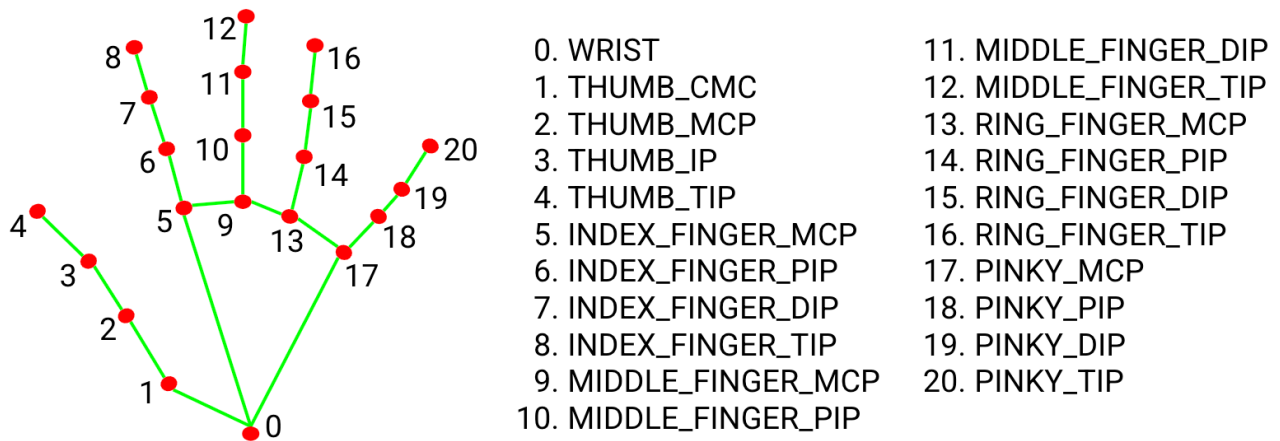


Figure 6. Arm joints shown on the MediaPipe skeletal model

In our model, we configured the sequence length to 20 frames per prediction, with each frame consisting of 126 input parameters. For model evaluation, we selected three gestures from the Kazakh sign alphabet: “А”, “Г”, and “Ғ”, represented by the Latin letters “A”, “G”, and “Gh” respectively, since the models only recognize the Latin alphabet. These gestures were chosen to ensure a comprehensive assessment of recognition accuracy by testing both distinct and similar gestures. The gestures “G” and “Gh” share the same hand position, but “Gh” additionally involves movement, classifying it as a dynamic gesture.

For training, we compiled datasets comprising 20 sequences per gesture and 10 sequences per hand. During the training process, each sequence was unrolled using the Backpropagation Through Time (BPTT) algorithm, where each element of the sequence represents a time step along the timeline. BPTT with LSTM operates as follows:

1. Forward pass: The LSTM processes the input sequence one time step at a time, updating the cell state and outputs at each time step.
2. Backward pass: Gradients of the loss function with respect to the output signal and cell state at each time step are computed using backpropagation through error.
3. Parameter update: The gradients are used to update the weights and biases of the LSTM using a standard optimization algorithm such as Stochastic Gradient Descent (SGD).
4. Repeat: This process is repeated for a specified number of epochs or until convergence.

During training, the LSTM uses Mean Squared Error (MSE) as the loss function and categorical accuracy as the accuracy metric. In the case of LSTM, the MSE loss function is used to measure the difference between the predicted output and the actual target output. During training, input sequences are fed into the LSTM one time step at a time, and the outputs at each time step are compared with the corresponding target outputs. The MSE loss is then calculated as the mean of the squares of the differences between the predicted output and the target output across all time steps of the sequence. The MSE loss function can be mathematically expressed as:

$$MSE = \frac{1}{N} * \sum (y_p - y_t)^2 \quad (4)$$

here:

N – number of time steps in the sequence;

y_p – predicted output of LSTM;

y_t – actual target output.

During training, the categorical accuracy metric is utilized to monitor the performance of LSTM and adjust network parameters using BPTT. The objective of training LSTM using categorical accuracy is to maximize the number of correctly classified sequences, enabling the net-

work to classify input sequences into different categories with high accuracy. The categorical accuracy metric can be mathematically expressed as:

$$CA = \frac{N_c}{N_t} \quad (5)$$

here:

N_c – number of correctly classified sequences;

N_t – total number of sequences.

After training, we tested the model on real examples. As expected, LSTM was able to recognize the gestures “A” and “G” with an accuracy of over 90%. However, it performed worse in recognizing the “Gh” gesture, often mistaking it for the “G” gesture. Additionally, the model demonstrated better recognition of gestures made with the left hand compared to the same gestures made with the right hand. This difference is likely due to the datasets for the left hand being better than those for the right hand. Nevertheless, the neural network performed admirably and justified the potential for dynamic gesture recognition.

A drawback of LSTM is that computing a sequence of data requires more time than computing individual data points, resulting in video delay. On average, gesture recognition by the trained neural network took 70-75 ms. Even with a standard frame rate of 25 frames per second, one frame takes 40 ms, meaning that there is approximately a 2x delay.

Long Short-Term Memory (LSTM) networks are a powerful type of recurrent neural network capable of processing long-term dependencies. In practice, they excel at recognizing dynamic gestures but experience delays in real-time recognition.

Data processing: The Kazakh sign alphabet comprises 42 gestures for each letter of the Kazakh alphabet. Since the model does not work with Cyrillic characters, each gesture was assigned a label in Latin transcription as numerical values ranging from 0 to 41 (see Table 3).

Table 3. Letter and number assigned to it.

A: 0	A': 1	B: 2	V: 3	G: 4	Gh: 5	D: 6
E: 7	E': 8	Zh: 9	Z: 10	I: 11	I': 12	K: 13
Q: 14	L: 15	M: 16	N: 17	N': 18	O: 19	O': 20
P: 21	R: 22	S: 23	T: 24	Y': 25	U: 26	U': 27
F: 28	X: 29	H: 30	C: 31	Ch: 32	Sh: 33	Sh': 34
Y: 35	I^: 36	Soft Sign: 37	Solid Sign: 38	Eh: 39	Iu: 40	Ia: 41

For each sign of the Kazakh sign alphabet, key points were extracted using the MediaPipe library. Key points are points that represent the position of the hands in the frame and relate to joints. Each hand has 21 key points, and each point has its coordinates (x, y, z). The MediaPipe library effectively captures anatomical points of the hands and converts them into numpy arrays, with special attention to the left and right hands. Each gesture is represented by a sequence of 10 numpy arrays, corresponding to every third frame from 0 to 27. In total, 20 sequences were collected for each sign, resulting in 840 sequences for all gestures.

Data Splitting: To evaluate the model's performance, the dataset was split into training and testing sets. 75% of the sequences (630) were allocated for training, providing the model with a significant amount of diverse training data. The remaining 25% of sequences (210) were reserved for testing, allowing for an objective assessment of the model's generalization ability on new, previously unseen data.

Model Training: A Long Short-Term Memory (LSTM) network was implemented using TensorFlow. According to Figure 4.16, the model consists of 3 LSTM layers and 3 dense layers. The

softmax activation function is applied to the output layer to create a probability distribution over 42 classes, representing each gesture of the Kazakh sign alphabet. The model is compiled using the Adam optimizer, categorical cross-entropy loss function, and categorical accuracy as the evaluation metric.

Model: "sequential"

Layer (type)	Output Shape	Param #
lstm (LSTM)	(None, 10, 16)	9152
lstm_1 (LSTM)	(None, 10, 32)	6272
lstm_2 (LSTM)	(None, 16)	3136
dense (Dense)	(None, 16)	272
dense_1 (Dense)	(None, 8)	136
dense_2 (Dense)	(None, 42)	378

=====
 Total params: 19,346
 Trainable params: 19,346
 Non-trainable params: 0

Figure 7. Model Layers

Table 4. Evaluation Metrics for the LSTM Model

Class	Precision	Recall	F1 score	Support
Ae(Ә)	0.85	0.62	0.78	19
Blank	0.81	0.91	0.29	8
Gh(Ғ)	0.88	0.64	0.78	15
Hh(Һ)	0.81	0.31	0.47	18
Ii(І)	0.81	0.76	0.89	7
Kk(Қ)	0.82	0.55	0.71	17
Nn(Ң)	0.81	0.92	0.91	15
Oe(Ө)	0.79	0.67	0.80	8
Ue(Ү)	0.79	0.91	0.89	6
Accuracy	0.82			79
Macro Avg	0.90	0.72	0.73	79
Weighted Avg	0.95	0.67	0.73	79

The Confusion Matrix provides a more detailed breakdown of the model's predictions and actual labels. It is organized into four categories: true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). Each row of the matrix corresponds to the actual class, and each column corresponds to the predicted class. Figure 4 shows the confusion matrix for the test data.

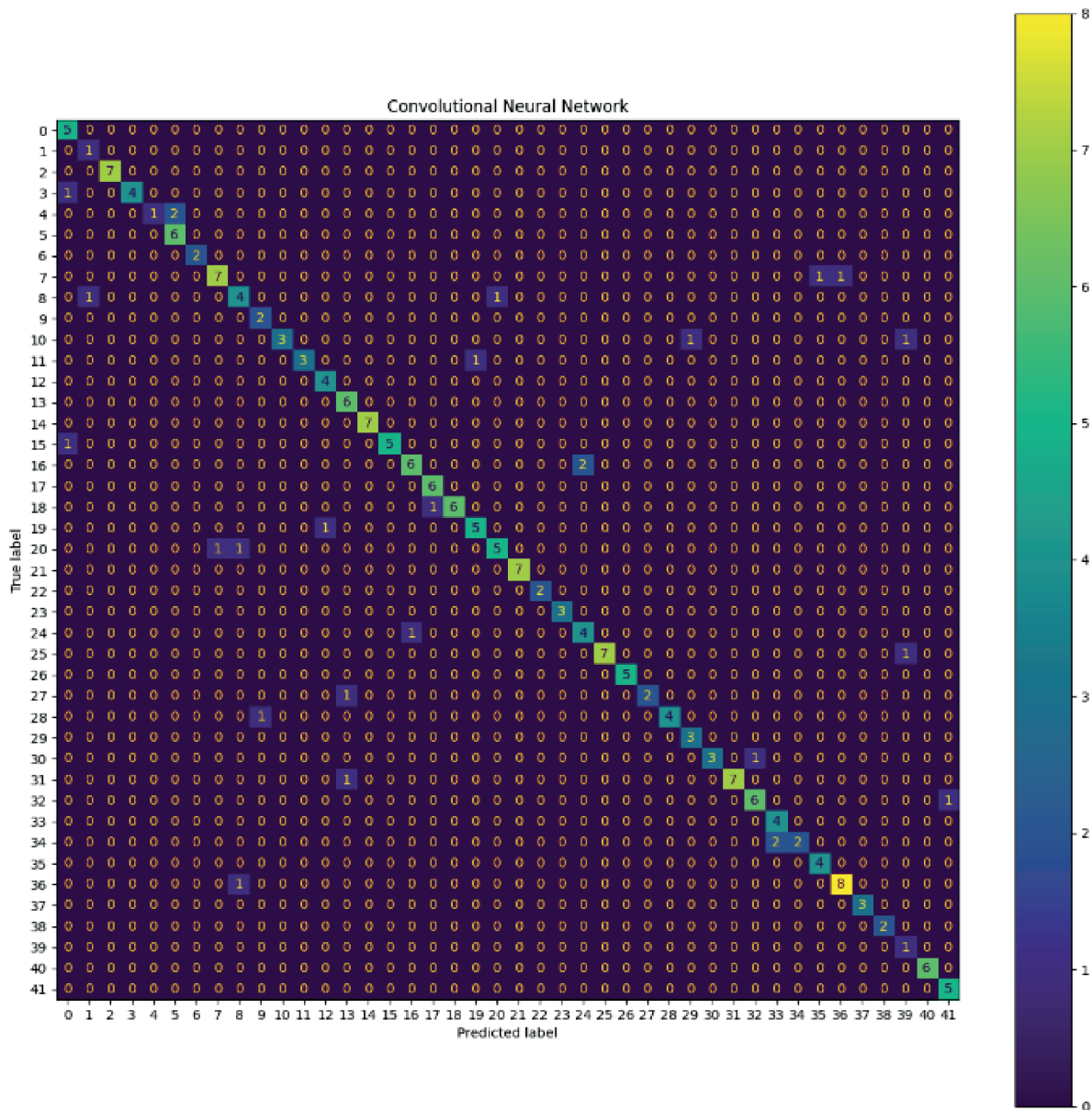


Figure 8. Confusion Matrix of LSTM

In Kazakh sign language, there are both static and dynamic gestures. These include pairs of gestures such as G and Gh, E and E', I and I', N and N', Y' and U, Sh and Sh'. Signs K and Q are similar, but Q requires slightly more movement than K. Signs C, soft sign, and hard sign are also dynamic but do not have similar gestures. In the confusion matrix, we observe that the model misclassified some Gh signs as G, N as N'(H), and Sh as Sh'(Ш). Therefore, we conclude that in most cases, the model can determine whether a gesture is static or dynamic, or if it is the same gesture but dynamic.

Convolutional Neural Network model

Convolutional neural networks (CNNs), with their ability to extract and learn intricate patterns from visual data, have emerged as powerful tools for this task. This article delves into a comparative analysis of two CNN architectures, 'model1' and 'model2', designed for hand gesture recognition.

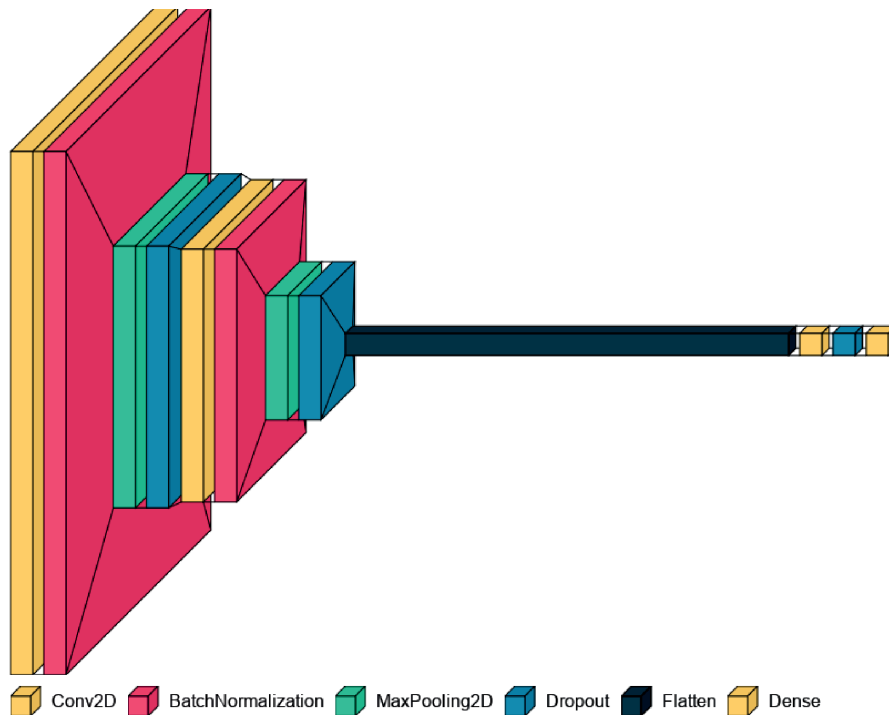


Figure 9. CNN Model 1: A Streamlined and Efficient Architecture

The presented sequential Convolutional Neural Network model is meticulously designed to proficiently capture and learn hierarchical features from input images. The model initiation involves a Conv2D layer employing 32 filters with a kernel size of (3, 3), yielding feature maps with dimensions (100, 120, 32). Following this, a BatchNormalization layer enhances model stability and convergence by normalizing activations, contributing to a robust training process. Subsequently, a MaxPooling2D layer reduces spatial dimensions by half, resulting in feature maps sized (50, 60, 32). To address overfitting, a Dropout layer with a strategically chosen rate of 0.25 is thoughtfully introduced.

The architectural depth is further augmented with a subsequent Conv2D layer, leveraging 64 filters to extract more intricate patterns. BatchNormalization is reapplied, followed by MaxPooling2D, which reduces spatial dimensions to (25, 30, 64). Another Dropout layer is introduced to further regularize the model. The subsequent Flatten layer transforms the 3D feature maps into a 1D vector with a size of 48,000.

The densely connected layers follow the flattened representation, commencing with a Dense layer comprising 128 units. To prevent overfitting in this dense layer, a Dropout layer with a rate of 0.25 is judiciously incorporated. The final Dense layer encompasses 10 units, employing softmax activation for multi-class classification.

In terms of parameters, the model boasts a total of 6,164,618 parameters, with 6,164,426 being trainable. These parameters encapsulate the weights and biases associated with both convolutional and dense layers. It's noteworthy that BatchNormalization layers contribute 192 non-trainable parameters. The architectural design of this sequential CNN is deliberately crafted to strike a harmonious balance between model complexity and interpretability, rendering it particularly well-suited for image classification tasks.

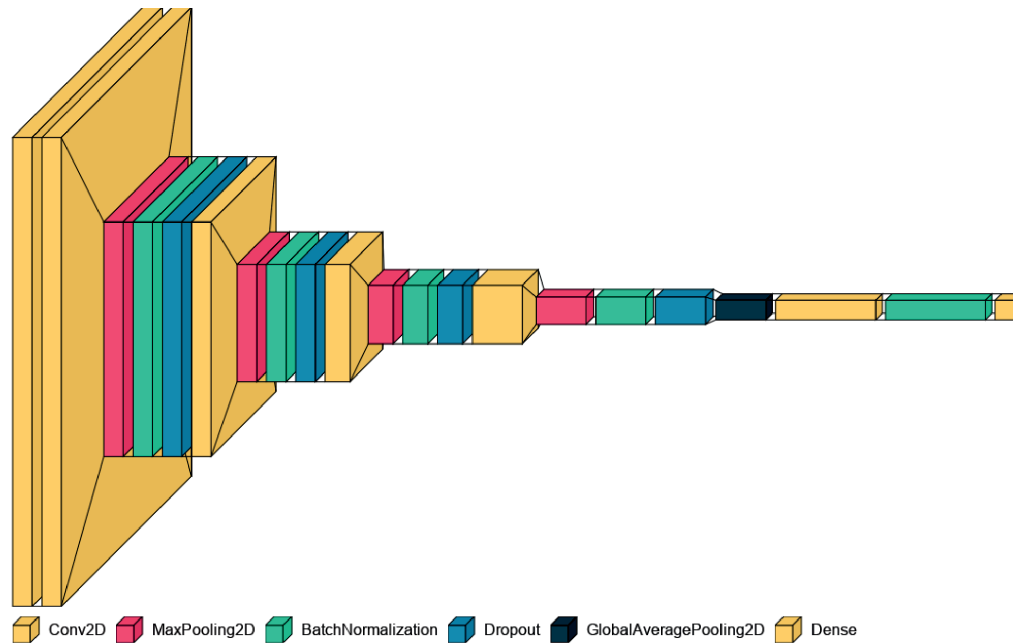


Figure 10. CNN Model 2: A Deep and Complex Architecture

The model is constructed as a sequential Convolutional Neural Network (CNN), meticulously designed for image classification tasks. The architecture unfolds with a Conv2D layer, denoted as 'Conv2D (32 filters)', utilizing 32 filters with a 3×3 kernel size to capture low-level image patterns. This initial layer contributes 320 trainable parameters. Following this, a second Conv2D layer, named 'Conv2D (64 filters)', further refines the feature representation with an increased number of filters, contributing an additional 18,496 trainable parameters. MaxPooling2D layers ('MaxPooling2D') succeed each convolutional layer, systematically reducing spatial dimensions while preserving crucial features. BatchNormalization layers ('BatchNormalization') are strategically placed after pooling operations, enhancing model stability and convergence. Dropout layers ('Dropout (0.25)') are incorporated to mitigate overfitting by randomly deactivating neurons during training.

The architectural depth is progressively expanded with additional convolutional layers, each incrementally increasing in the number of filters ('Conv2D (128 filters)', 'Conv2D (256 filters)', 'Conv2D (512 filters)'). MaxPooling, BatchNormalization, and Dropout layers are consistently integrated, maintaining a delicate balance between model complexity and generalization. Post the convolutional layers, a GlobalAveragePooling2D layer ('GlobalAveragePooling2D') is employed to condense spatial dimensions to a single vector, effectively capturing the global information of the learned features.

Transitioning to fully connected layers, the model incorporates a Dense layer ('Dense (1024 units)') with rectified linear unit (ReLU) activation, facilitating the transformation of the extracted features into a high-level representation. This dense layer contributes 5,254,144 trainable parameters. BatchNormalization is again applied for regularization, adding 4,096 trainable parameters. The final layer, denoted as 'Dense (10 units, Softmax)', employs softmax activation for multi-class classification, generating probability distributions over the target classes. This comprehensive architecture, with a total of 2,111,498 trainable parameters, is carefully designed to strike a harmonious balance between depth and complexity, empowering the model to discern intricate patterns within images and provide accurate predictions across diverse classes.

Comparative Analysis: Unveiling the Strengths and Trade-offs

Architectural Depth and Complexity. A fundamental distinction between the two models lies in the depth and complexity of their architectures. The first model boasts a more profound structure, featuring multiple convolutional layers (Conv2D), BatchNormalization, MaxPooling2D, Dropout layers, and a GlobalAveragePooling2D layer. This intricate arrangement enables the model to extract and process intricate patterns within hand gestures, potentially leading to enhanced recognition accuracy. In contrast, the second model adopts a shallower architecture with fewer layers, encompassing Conv2D, BatchNormalization, MaxPooling2D, Dropout, and Dense layers. While this simpler design may compromise feature extraction capabilities, it offers the advantage of reduced computational complexity and faster processing times.

Pooling Techniques for Feature Selection and Aggregation. Pooling techniques play a significant role in feature selection and aggregation, influencing the models' ability to extract and represent relevant information from hand gesture images. The first model employs MaxPooling2D layers after each convolutional layer, emphasizing feature selection by retaining the most prominent features while discarding less informative ones. In contrast, the second model combines MaxPooling2D layers after convolutional layers with GlobalAveragePooling2D. This suggests a different approach to feature aggregation, potentially capturing more global information from the entire hand gesture image.

Resource Efficiency Considerations. The choice between the two models becomes particularly crucial when considering resource efficiency. The first model, due to its deeper architecture and higher parameter count of 6,164,618, demands substantial computational resources. This may limit its applicability in environments with constrained processing power. Conversely, the second model, with its significantly smaller parameter count of 2,111,498, proves to be more resource efficient. This characteristic makes it well-suited for deployment on embedded systems or devices with limited computational capabilities.

Capacity for Feature Representation. The capacity for feature representation, determined by the depth and complexity of the network architecture, directly impacts the models' ability to capture intricate details and patterns within hand gestures. The first model, with its deeper architecture, exhibits higher capacity, potentially enabling the capture of more subtle features that contribute to accurate gesture recognition. Conversely, the second model, with its shallower architecture, has a lower capacity for feature representation. This may limit its ability to discern fine-grained details, potentially affecting gesture recognition accuracy in situations where subtle hand movements are crucial.

Learning Rate and Dropout Optimization. Learning rate and Dropout optimization strategies play a crucial role in tailoring the learning process and preventing overfitting. The first model utilizes the Adam optimizer and specifies Dropout rates at different points in its architecture. This approach may provide more flexibility in adjusting Dropout rates for the first model's shallower architecture. In contrast, the second model employs the Nadam optimizer with a learning rate of 0.005 and incorporates Dropout with a rate of 0.25. These settings may optimize the learning process for the second model's deeper architecture.

Regularization Techniques for Overfitting Prevention. Both models integrate Dropout layers to combat overfitting, a common issue in machine learning that can lead to poor generalization performance. The first model strategically deploys Dropout layers after both convolutional and dense layers, effectively reducing the influence of overly dominant features. The second model, on the other hand, incorporates Dropout at various points throughout its architecture, targeting potential overfitting at multiple stages of the learning process. This approach may provide a more comprehensive regularization strategy.

Training and Validation loss

Model 1 starts with a relatively high training loss of 4.65, indicating that the initial predictions are far from the ground truth. However, over subsequent epochs, the model shows a consistent downward trend in the training loss. This reduction suggests that the model is effectively learning from the training data and making improvements. In contrast to the training loss, the validation loss for Model 1 exhibits more fluctuations. Although it follows a decreasing trend, the noticeable discrepancy between the training and validation loss suggests a potential problem of overfitting or lack of generalization. The increasing gap between the two losses indicates that Model 1 may be performing well on the training data but struggles to generalize to unseen examples. Model 2 begins with a higher training loss of 2.83, indicating a relatively poorer initial performance compared to Model 1. However, the model demonstrates a steady decline in training loss over the epochs. The convergence of the training loss suggests that Model 2 effectively learns from the training data and improves its predictions. Like Model 1, Model 2 shows fluctuations in the validation loss but follows a decreasing trend overall. However, the gap between the training and validation loss is less pronounced compared to Model 1. This suggests that Model 2 achieves better generalization capabilities, performing well not only on the training data but also on unseen examples.

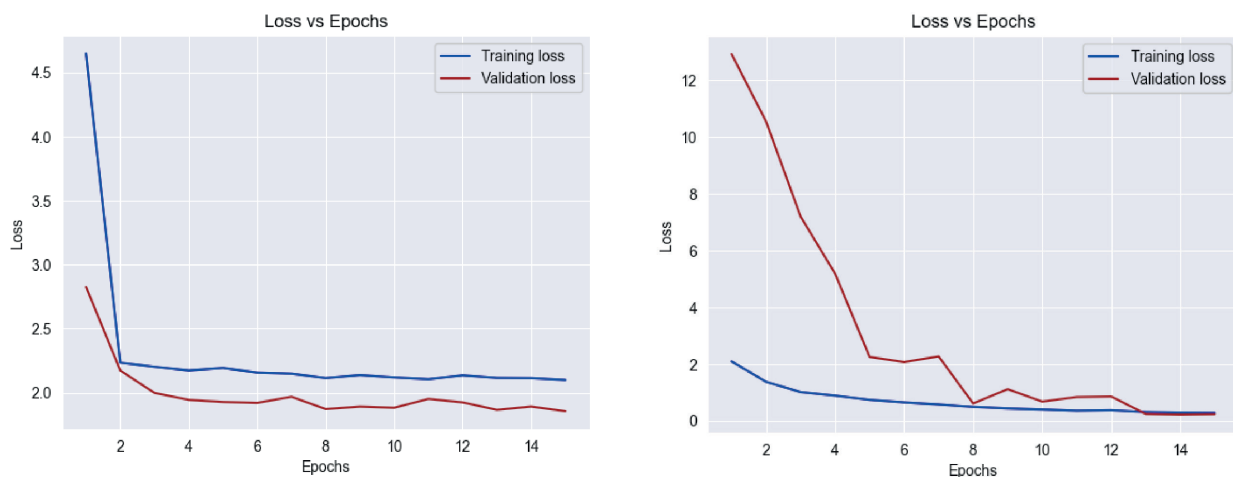


Figure 11. Comparing CNN models

Both models show a decreasing trend in training and validation loss, improving over time. Model 2 shows better convergence, with higher initial loss but lower final loss, suggesting better learning and generalization capabilities. Model 1 shows overfitting and lack of generalization.

Model Training Process Analysis - Training and Validation Accuracy

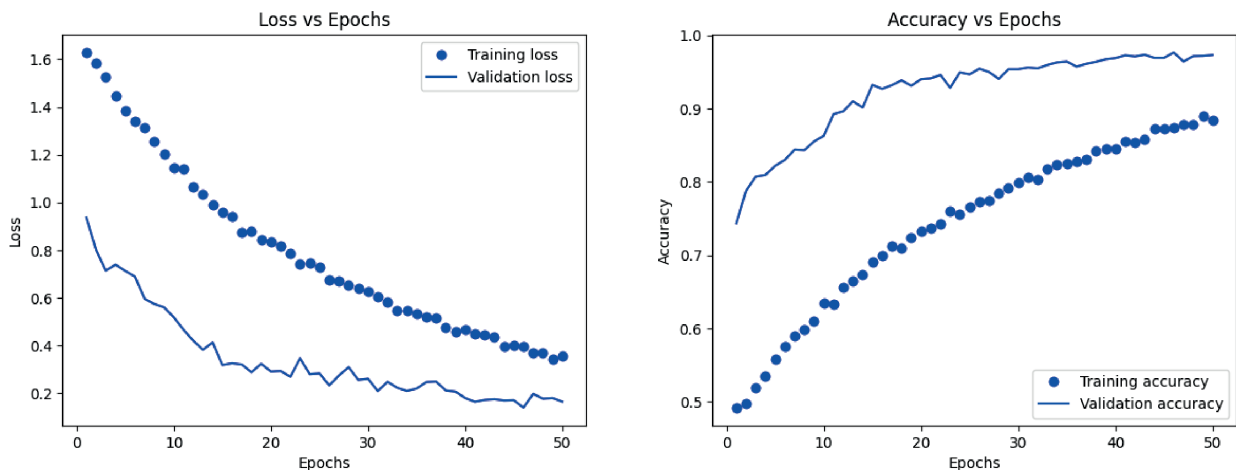


Figure 12. Loss and epoch of CNN models

Model 1 starts with a relatively low training accuracy of 18.66%. However, it shows consistent improvement over the epochs, indicating that the model learns from the training data and adjusts its predictions accordingly.

The validation accuracy for Model 1 begins at 18.0% and shows a positive trend over the epochs. However, the fluctuations in the validation accuracy suggest challenges in generalizing the model's predictions to unseen data.

Model 2 starts with a higher training accuracy of 34.9% and steadily increases over the epochs. This indicates that the model effectively learns from the training data and improves its accuracy.

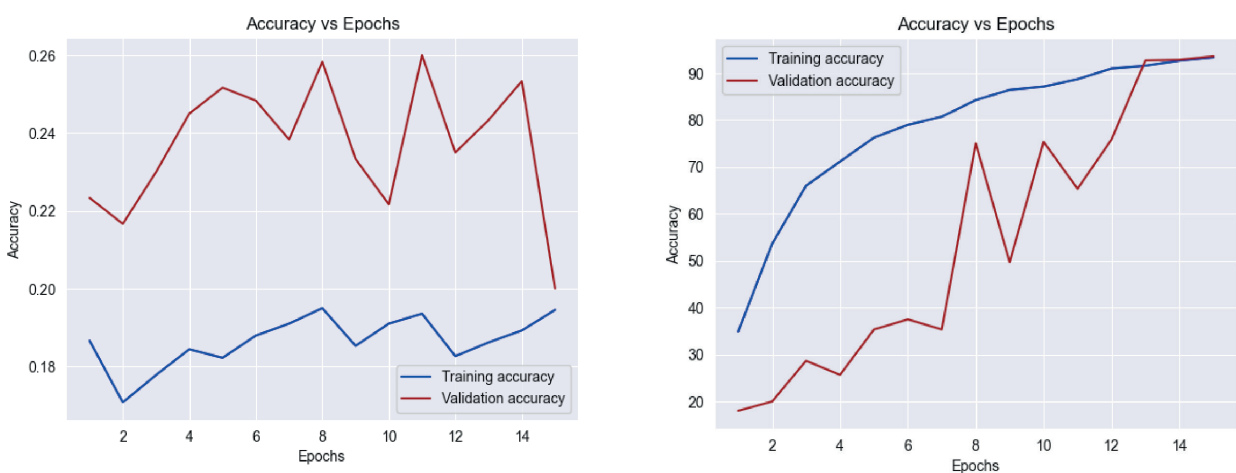


Figure 13. Loss and Epoch of CNN models

The validation accuracy for Model 2 begins at 20.0% and follows a positive trend. The close alignment between the training and validation accuracy suggests that Model 2 generalizes well to unseen data. Model 2 exhibits higher initial training accuracy and validation accuracy compared to Model 1. Both models show an improving trend in accuracy over the epochs, indicating effective learning from the training data. However, Model 2 demonstrates better generalization capabilities, as evidenced by its higher validation accuracy and closer alignment between training and validation accuracy.

Figure 14 shows the overall performance graph of all models, including the proposed model based on a hybrid architecture using layers of recurrent and convolutional neural networks (LSTM + MediaPipe). The recognition accuracy of Kazakh gestures exceeds 95%.

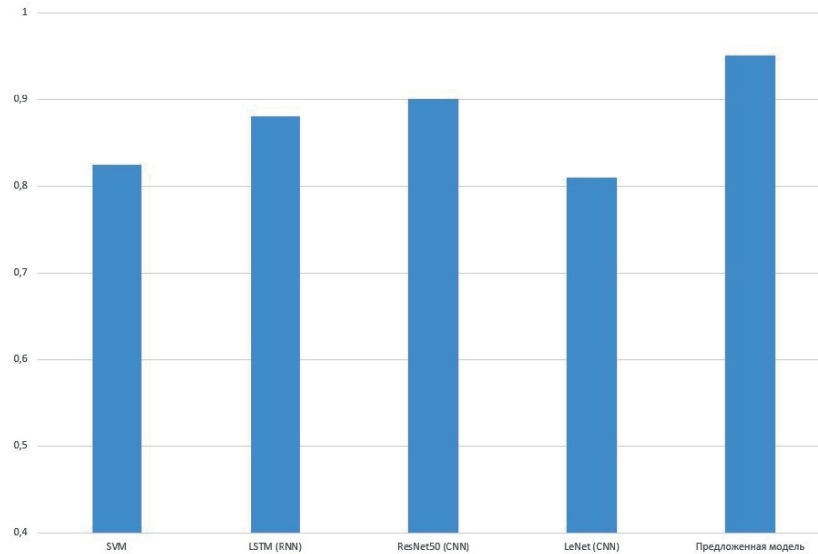


Figure 14. Overall Recognition Accuracy of Models (Histogram)

Experiments have demonstrated that the accuracy of recognizing Kazakh letters, and the Kazakh sign language alphabet in general, is validated by calculations during training, testing, and the justification of results according to the applied metrics described and applied above, as well as the visualization of the confusion matrix. Therefore, it can be concluded that the proposed hybrid architecture model is more accurate compared to other machine learning and deep learning neural network models.

Testing models by application

Real-time gesture recognition is a process that allows computers and other devices to recognize and interpret gestures made by a person in real time. The developed software offers the user to recognize hand gestures of people with disabilities. The main purpose of the software is to invite the consumer to interact and build communication with people who cannot speak due to any impairment. Below is a demo of the application.

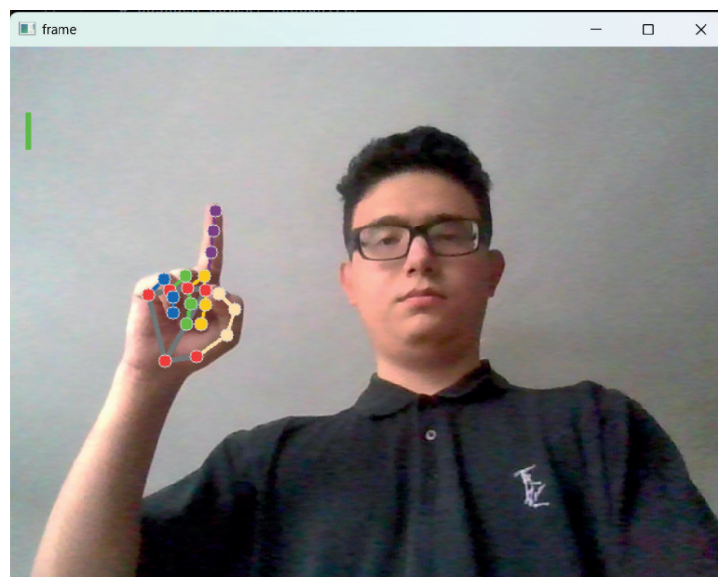


Figure 15. Recognition of the letter "I"

The image above shows the results of recognizing the next letter:



Figure 16. Recognition of the letter “A”

The SVM recognition results are shown in Figure 16.

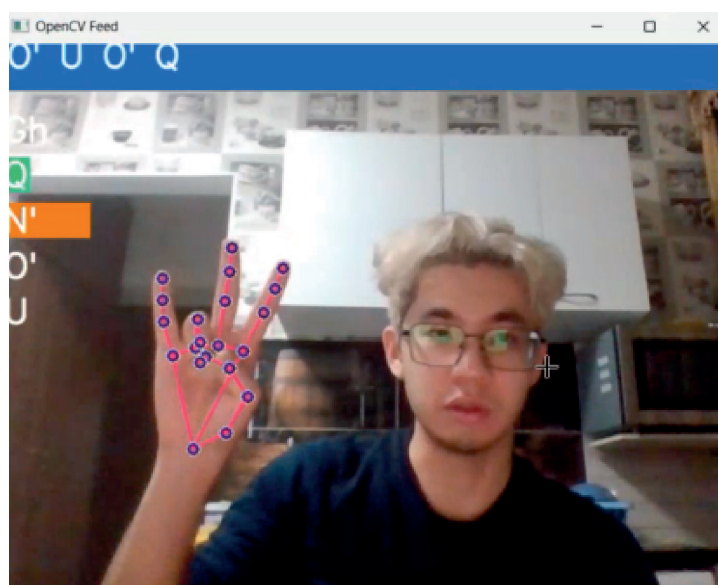


Figure 17. Recognition of the letter “H”

Figure 17 shows the recognition results using the Long-Short Term Memory recurrent neural network.

Conclusion

Concluding this research work we got result of training methods and models of recurrent and neural network for gesture recognition of Kazakh Sign Language. In this work explored machine learning methods and pattern recognition algorithms, specifically for gestures. This study was dedicated to advancing and implementing highly accurate pattern recognition methodologies. Specifically, images depicting hand gestures from the Kazakh sign alphabet served as the primary data input.

Throughout the investigation, a comprehensive examination of literature focusing on pattern recognition, particularly hand gestures, was conducted. Each of the referenced studies employed distinct empirical approaches tailored to their specific and narrow research contexts.

The study delved into various machine learning and deep learning frameworks, including convolutional and recurrent neural networks, utilizing SVM, CNN (supervised learning) and LSTM (deep learning) models for sequential data processing. These methodologies demonstrated robust training and testing outcomes, as measured by metrics such as F1-score, accuracy, precision, recall, and confusion matrix.

A proprietary hybrid model was developed to address the intricate multi-class classification challenge. The integration of LSTM and CNN layers exhibited the efficacy of the proposed methodology in Kazakh letter recognition. The artificial neural network achieved gesture recognition rates exceeding 95% probability, as assessed by the applied metrics, and reliably classified them.

Nevertheless, significant challenges persist, including the necessity for substantial volumes of annotated data. Current resources of reference images of hand gestures from Kazakh sign language and sign language translations are insufficient. Additionally, the demand for extensive computational resources to accommodate escalating data volumes and increasingly complex model architectures poses another obstacle.

Consequently, the theoretical, practical, and experimental research conducted constitutes a comprehensive dissertation cycle, encompassing the acquisition of results and their validation through presentation at conferences and dissemination to relevant organizations.

References

- [1] Mukhanov, S.B., & Uskenbayeva, R.K. (2020). Pattern Recognition with Using Effective Algorithms and Methods of Computer Vision Library. *Advances in Intelligent Systems and Computing*, Article 991, 810-819. https://doi.org/10.1007/978-3-030-21803-4_81
- [2] Mukhanov, S., Uskenbayeva, R., Young, I.Ch., KabyL, D., Les, N., & Amangeldi, M. (2023). Gesture Recognition of Machine Learning and Convolutional Neural Network Methods for Kazakh Sign Language. *Scientific Journal of Astana IT University*. 15(15), 85–100. <https://doi.org/10.37943/15LP-CU4095>
- [3] Amirgaliev, E.N., Mukhanov, S.B., Zheksenov, D.B., Kalzhigitov, N.K., Lee, A.S., Evdokimov, D.D., & Kenshimov, C. (2023) A comparative analysis of neural network models for hand gesture recognition methods. *Bulletin of the National Engineering Academy of the Republic of Kazakhstan*. 2(88), 15-27. <https://doi.org/10.47533/2023.1606-146X.2>
- [4] 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://doi.org/10.15587/1729-4061.2021.241535>
- [5] Aitulen, A.D., & Mukhanov, S.B. (2019). Processing, identification and recognition by Viola-Jones method. *VESTNIK KazNRTU*. 6(136), 155-161.
- [6] Uskenbayeva, R.K., & Mukhanov S.B. (2020). Contour analysis of external images. *Proceeding of the ACM International Conference Proceeding Series*, Article 3410811. <https://doi.org/10.1145/3410352.3410811>
- [7] Bazarevsky, V., & Fan, Zh. (2019). On-device, real-time hand tracking with mediapipe. Google AI Blog. Available at: <https://ai.googleblog.com/2019/08/on-device-real-time-hand-tracking-with.html>.
- [8] 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-razrabotkoykazakhstanstantsev-dlya-glukhikh.html>
- [9] Bazarevsky, V., & Fan Zh. (2019, August 19). 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>.
- [10] 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>.

- [11] 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>.
- [12] Bilgin, M., & Mutludogan, K. (2019). American Sign Language character recognition with capsule networks. *Proceedings of the 3rd International Symposium on Multidisciplinary Studies and Innovative Technologies, Ankara, Turkey*. <https://doi.org/10.1109/ismsit.2019.8932829>.
- [13] Kudubaeva, S.A., Ryumin, D.A. & Kalzhanov M.U. (2016). Support vector machine for sign speech recognition using the KINECT sensor. *Bulletin of KazNU. Series "Mathematics, mechanics, computer science"*. 91(3). <https://bm.kaznu.kz/index.php/kaznu/article/view/541>
- [14] 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>.
- [15] Lai, K., & Yanushkevich, S.N. (2018). CNN+RNN depth and skeleton based dynamic hand gesture recognition. *Proceeding of the 24th International Conference on Pattern Recognition (ICPR), IEEE*. <https://doi.org/10.1109/ICPR.2018.8545718>
- [16] 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>
- [17] 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, Lecce, Italy*, 47-57, 24-27. https://doi.org/10.1007/978-3-319-48680-2_5.
- [18] Kumar, A., Thankachan, K., & Dominic, M.M. (2016) Sign language recognition. *Proceedings of the 3rd IEEE international conference on recent advances in information technology (RAIT)*, 422–428. <https://doi.org/10.1109/rait.2016.7507939>.
- [19] Tan, M., & Le, Q. (2019). Efficientnet: Rethinking model scaling for convolutional neural networks. *Proceedings of the International Conference on Machine Learning*, 6105-6114. <https://arxiv.org/abs/1905.11946>.
- [20] Lau, S., Gonzaltz, J., & Nolan, D. (2023). *Learning Data Science*. O'Reilly Media, Inc. 596.
- [21] McKinney, W. (2022). *Python for Data Analysis: Data Wrangling with pandas, NumPy, and Jupyter* (3rd ed.) O'Reilly Media.
- [22] Merembayev, T., Kurmangaliyev, D., Bekbauov, B., & Amanbek, Y. (2021). Comparison of machine learning algorithms in predicting lithofacies: Case studies from Norway and Kazakhstan. *Energies*. 14(7), 1896.
- [23] Zhang, Y., Cao, C., Cheng, J., & Lu, H. (2018). EgoGesture: A New Dataset and Benchmark for Ego-centric Hand Gesture Recognition. *IEEE Transactions on Multimedia*. 20(5). <https://doi.org/10.1109/TMM.2018.2808769>
- [24] Shi, W., Cao, J., Zhang, Q., Li, Y., & Xu, L. (2016). Edge computing: Vision and challenges. *IEEE internet of things journal*. 3(5), 637-646. <https://doi.org/10.1109/JIOT.2016.2579198>
- [25] Wong, B.P., & Kerkez, B. (2016). Real-time environmental sensor data: An application to water quality using web services. *Environmental Modelling & Software*. 84, 505-517. <https://doi.org/10.1016/j.envsoft.2016.07.020>
- [26] Granell, C., Havlik, D., Schade, S., Sabeur, Z., Delaney, C., Pielorz, J., & Mon, J.L. (2016). Future Internet technologies for environmental applications. *Environmental Modelling & Software*. 78, 1-15.
- [27] Alvarez, M.A., & Lawrence, N.D. (2011). Computationally efficient convolved multiple output Gaussian processes. *The Journal of Machine Learning Research*. 12, 1459-1500.
- [28] Futoma, J., Hariharan, S., & Heller, K. (2017). Learning to detect sepsis with a multitask Gaussian process RNN classifier. *Proceedings of the International conference on machine learning (PMLR)*.1, 1174-1182.
- [29] Elman, A., & Hill, J. (2006). *Data analysis using regression multilevel/hierarchical models*. Cambridge university press. 122.
- [30] 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>
- [31] 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>.