

National Research University Higher School of Economics
Faculty of Computer Science
Programme 'Master of Data Science'

MASTER'S THESIS
Russian Sign Language Dactyl Recognition from Video in
Dactyl Education Application

Student:

Olga Vasileva

Supervisor:

PhD, Associate Professor, Ilya Makarov

Moscow, 2022

ABSTRACT

Sign language is the main way for deaf people and people with hearing impairment to communicate between each other and with family members who do not have these disabilities.

Most people do not know sign language, so the automation of sign language interpretation is one of the hot topics, progress in this area will simplify communication between people with hearing impairments and people without them.

This paper focuses on the recognition of Russian sign language dactyls. There are papers on this topic with good results on recognition of static dactyls and some dynamic ones, represented as static, using deep convolutional neural networks. In this master's thesis, which is a group project created by students Tsimafei Akulich and Olga Vasileva, we have received more accurate results on static dactyls recognition and implemented dynamic dactyl recognition from video. In our approach, we use hand pose estimation, the author's algorithm for feature generation that considers the anatomical characteristics of a hand, and one of the classical machine learning algorithms for classification.

To train the model we created our own dataset consisting of more than 66 thousand images. To make our work complete, we created the dactyl education application with an interactive mode for recognizing the dactyl shown by the user.

TABLE OF CONTENTS

DEFINITIONS, DESIGNATIONS AND ABBREVIATIONS.....	4
INTRODUCTION.....	5
GENERAL DESCRIPTION OF A TEAM PROJECT	7
CHAPTER 1. OVERVIEW OF AVAILABLE RESULTS AND CURRENT APPROACHES.....	9
CHAPTER 2. DATASET.....	13
2.1 Russian dactyl alphabet.....	13
2.2 Dataset concept.....	14
2.3 Train dataset creation.....	14
CHAPTER 3. PIPELINE.....	16
3.1 Hand pose estimation and landmarks	16
3.2 Feature generation.....	17
3.3 Model selection for dactyls classification.....	22
3.4 Results comparison and selection of the best model.....	25
CHAPTER 4. TESTING THE MODELS UNDER DIFFERENT CONDITIONS	27
4.1 Distance	27
4.2 Illumination level.....	28
4.3 Rotation	30
4.4 Color of clothes (background).....	31
CHAPTER 5. APPLICATION	33
5.1 Application description.....	33
5.2 Application flow	33
5.3 Application operation modes	34
5.3.1 Simple detection and classification mode.....	34
5.3.2 Correct dactyl mode.....	35
5.3.3 Word guessing mode	36
CONCLUSION	37
REFERENCES.....	38
APPLICATIONS	40

DEFINITIONS, DESIGNATIONS AND ABBREVIATIONS

RSL — Russian sign language

SL — sign language

RDA — Russian dactyl alphabet

ML — machine learning

SDK — software development kit

INTRODUCTION

About 13 million people in Russia have some degree of hearing loss. The degree of hearing loss can range from mild hearing loss to total deafness. Learning the Russian sign language (RSL) begins with learning the Russian dactyl alphabet (RDA).

Children with congenital hearing loss learn the basics of grammar by spelling entire words using RDA before they even learn to write. Not only hearing-impaired people, but their relatives as well learn RSL, it allows them to maintain communication.

The automation of sign language (SL) interpretation is one of the actual topics, progress in this area will simplify communication between people with hearing impairments and people without them.

The dactyl recognition process is a standard machine learning pipeline: collecting and preprocessing data, splitting into samples, training the model, and examining the quality of the model on test data, the task is a multiclass classification. At this time, the available approaches using deep convolutional neural networks allow to recognize static RSL dactyls, the maximum obtained overall accuracy is 78%.

In our project the recognition of dactyls, both statistic and dynamic, is implemented with a fairly high accuracy, the overall accuracy on the test sample is 95.75%. In order to get good results, first we created a dataset ourselves, because the available datasets contain a small number of images, the quality of some images is low. The training sample consists of more than 47 thousand images, the test sample consists of almost 19 thousand images. Five people participated in the creation of the training sample, two people participated in the creation of the test sample, and four types of augmentations were additionally used for each image.

For hand pose estimation we have used MediaPipe Hands, which gives the coordinates of 21 points in three-dimensional space, each point corresponds to an anatomical formation of the hand. Some points are connected by bones or by almost immobile ligaments, which means that the distance between them does not change at all or not significantly. Other points have a joint between them, it means that the

distance between them varies with a movement. There are joints in the hand which movements are possible along all the axes, and joints in which movements are performed along one axis, that is, points that change positions independently of each other and points that can only change position together and in the same direction. We took these anatomical features into account and measured the distances between those points where appropriate. Then we normalized the distances obtained, so we got 35 features with values from 0 to 1. We took into account the position of the hand in space, such as the position of the hand with the fingers up or down and added another six features. Thus, we vectorized the initial images and obtained 41 features for each observation. We trained the Logistic Regression model on the obtained training sample, pre-selecting the best hyperparameters using a search with cross-validation.

We use our trained model in the application, getting real-time class predictions for images from a webcam. The application works as follows: the webcam shoots K frames per second, the frames are stored for S seconds, a queue of $K * S$ images is formed, the images are sent to MediaPipe Hands, the output (21 point coordinates) is processed by the algorithm that creates 41 features, the obtained data is sent to the trained Logistic Regression model, the output is a prediction of the class and the probability of that class, returned to the application. If the probability is greater than a certain threshold (0.9), the image is finally verified as a definite dactyl, this step eliminates the definition of arbitrary gestures as dactyls. The result of described work is a list of predictions of size $K * S$. The most frequent class is taken from the queue, so the probability of a random recognition is reduced. In addition, the coordinates of the index fingertip are sequentially recorded for each image, which makes it possible to determine the correct trajectory of a movement in case of dynamic dactyls recognition.

GENERAL DESCRIPTION OF A TEAM PROJECT

This project required creating a dataset on which to train the model, creating an image vectorization algorithm with pose estimation and an algorithm for finding distances between points, choosing a suitable model for multiclass classification, and developing an application.

This project can be divided into the following tasks:

1. Collecting dataset:
 - 1.1. Developing the concept of a dataset consisting of images, considering the representation of each dynamic dactyl in form of several images. Python script for image capture from a webcam.
 - 1.2. Collecting images of five participants for train dataset.
 - 1.3. Collecting images of two participants for test dataset.
2. Image vectorization:
 - 2.1. Hand pose estimation: python script for getting landmarks from images.
 - 2.2. Development of an algorithm to determine the distance between the required points and python script for feature generation.
3. Multiclass classification, selecting suitable models and finding the best hyperparameters for them, selecting the best model:
 - 3.1. Logistic Regression
 - 3.2. Decision Tree Classifier
 - 3.3. Random Forest Classifier
 - 3.4. C-Support Vector Classifier
 - 3.5. LightGBM Classifier
 - 3.6. Multilayer Perceptron with two hidden layers
4. Testing the best model under different conditions:
 - 4.1. Distance
 - 4.2. Illumination level

- 4.3. Rotation
- 4.4. Color of clothes (background)
- 5. Application development:
 - 5.1. Architecture development
 - 5.2. Simple detection and classification mode development
 - 5.3. Correct dactyl mode development
 - 5.4. Word guessing mode development

This group project was created by students Tsimafei Akulich and Olga Vasileva.
We have divided the above listed tasks as follows:

Tsimafei Akulich's tasks:

- 1.1, 1.3
- 2.1
- 3.1, 3.2, 3.6
- 4.3
- 5.1, 5.2, 5.3

Olga Vasileva's tasks:

- 1.2
- 2.2
- 3.3, 3.4, 3.5
- 4.1, 4.2, 4.4
- 5.4

CHAPTER 1. OVERVIEW OF AVAILABLE RESULTS AND CURRENT APPROACHES

Ilya Makarov, Nikolay Veldyaykin, Maxim Chertkov and Aleksei Pokoev studied the topic of recognition of the Russian static dactyls and some dynamic ones, represented as static, wrote three articles: American and Russian Sign Language Dactyl Recognition [1], Russian Sign Language Dactyl Recognition [2], American and Russian Sign Language Dactyl Recognition and Text2Sign Translation [3]. These articles describe studies in which the authors created two convolutional neural networks and used two convolutional neural networks created for American Sign Language recognition by other researchers [4][5].

The first model takes 32×32 RGB images as an input, the model is like the popular LeNet architecture [6], has shown high performance in the task of image classification, the architecture of the model is shown in Figure 1. There are two convolutional layers with 5×5 kernel and ReLU activation function, followed by MaxPooling layer with filters of size 2×2 applied with a stride of 2. The resulting feature maps are flattened, followed by 2 dense layers with ReLU activation and a SoftMax layer for the obtaining predictions for each class.

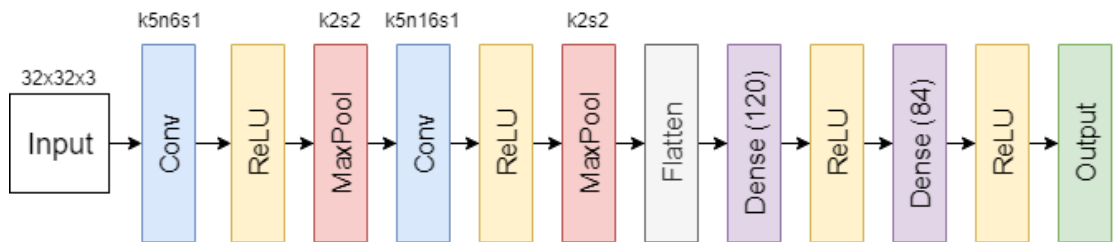


Figure 1 — [LeCun, 1998] model

The second model takes 32×32 images in gray colors in 1 channel as an input, it has a more complex architecture, which the authors called "QuadroConvPoolNet", the architecture of the model is shown in Figure 2. This median model without fully connected layer is similar to [7], but with a later position of pooling and different filters.

The convolutional kernels, pooling layer 3-1-1-4-5 and all the convolutional layers use a stride of 1, the model takes about 1 MegaByte in memory.

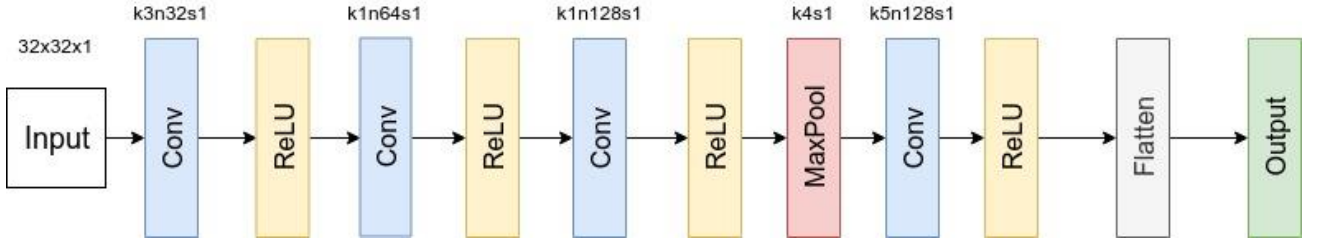


Figure 2 — *QuadroConvPoolNet*

The third model was created by Murat Taskiran, Mehmet Killioglu and Nihan Kahraman [4], the architecture of the model is shown in Figure 3. It takes 28×28 grayscale images as an input, the structure is like the first model, but convolutional layers have 3×3 kernel and the only one MaxPooling and one Dense layer.

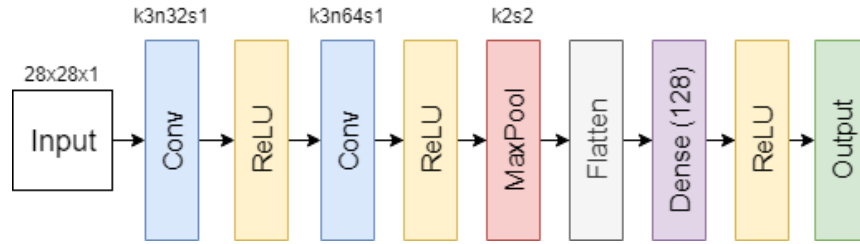


Figure 3 — *[Taskiran et al. 2018] model*

The fourth model was created by Debasrita Chakraborty, Deepankar Garg, Ashish Ghosh and Jonathan H. Chan [5], the architecture of the model is shown in Figure 4. It has 4 convolutional layers with 1×1 kernels. Each of them, except the third one, is followed by a MaxPooling layer. At the end of the network features are flattened and SoftMax function is applied to predict class. Before training, the model was trained as autoencoder for finetuning weights for the Encoder part.

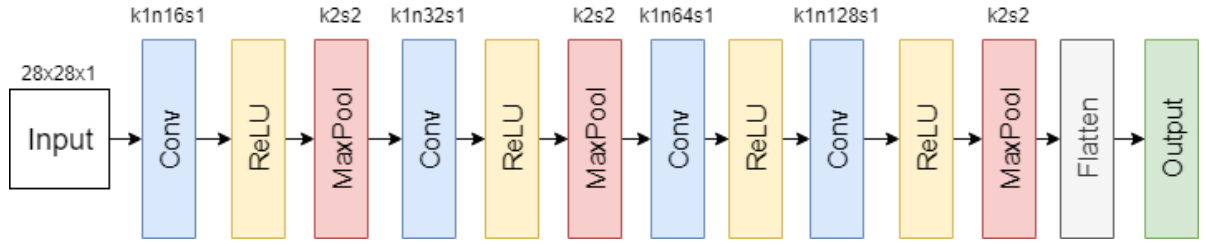


Figure 4 — [Chakraborty et al. 2018] model

Table 1 resents the descriptive statistics of the results obtained and Table 2 presents the prediction accuracies (%) of static dactyls on the test sample for the four models. In our paper we use the transliteration of the letters of the Russian alphabet according to GOST-7.79-2000 (system B) [8]. The QuadroConvPoolNet can be considered as the best model, testing this model on the test data set gave the best results. This model has the highest minimum accuracy value, the highest maximum accuracy value, the highest overall accuracy, and the smallest standard deviation.

Table 1 — Descriptive statistics of the obtained prediction accuracies

	[LeCun, 1998] model	QuadroConvPoolNet model	[Taskiran et al. 2018] model	[Chakraborty et al. 2018] model
Count	27	27	27	27
Mean	70.26%	77.56%	70.7%	33.59%
Std	13.02%	8.82%	10.97%	11.2%
Min	25%	61%	50%	20%
25%	65%	69%	60%	25.5%
50%	73%	80%	70%	32%
75%	78.5%	84.5%	81%	40%
Max	89%	93%	87%	65%

Table 2 — Accuracy on RSL test data

Letter	Transliteration	Transcription in the articles	[LeCun, 1998] model	QuadroConvPoolNet model	[Taskiran et al. 2018] model	[Chakraborty et al. 2018] model
A	A	a	51%	70%	60%	20%
Б	B	b	65%	75%	68%	22%
B	V	v	75%	86%	70%	27%
Г	G	g	68%	84%	59%	39%
Д	D	-	-	-	-	-
Е	Ye	e	75%	69%	68%	40%
Ё	Yo	-	-	-	-	-
Ж	Zh	zh	70%	83%	71%	28%
З	Z	-	-	-	-	-
И	I	i	83%	69%	78%	40%
Й	J	-	-	-	-	-
К	K	k	83%	86%	84%	33%
Л	L	l	65%	78%	81%	29%
М	M	m	78%	77%	80%	27%
Н	N	n	76%	82%	60%	24%
О	O	o	79%	80%	81%	21%
П	P	p	65%	61%	60%	20%
Р	R	r	67%	88%	72%	45%
С	S	s	89%	93%	87%	35%
Т	T	t	68%	69%	65%	65%
У	U	u	82%	88%	83%	40%
Ф	F	f	85%	82%	85%	43%
Х	X	kh	55%	80%	60%	45%
Ц	C	c	25%	87%	50%	25%
Ч	Ch	ch	82%	85%	85%	43%
Ш	Sh	sh	62%	62%	60%	55%
Щ	Shh	-	-	-	-	-
Ъ	``	-	-	-	-	-
Ы	Y`	yi	73%	65%	70%	26%
Ь	`	-	-	-	-	-
Э	E`	ye	58%	67%	50%	20%
Ю	Yu	yu	69%	69%	68%	30%
Я	Ya	ya	76%	81%	83%	32%
Overall			73%	78%	71%	33%
































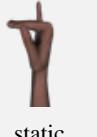

CHAPTER 2. DATASET

For our project, we needed a large RDA dataset that considers all the features of the recognition algorithm in the application, so we created it ourselves.

2.1 Russian dactyl alphabet

RDA consists of 33 dactyl signs, 24 of which are static and 9 are dynamic. Some dactyls have similarities with the printed characters (G, L, M, O, S), others are reproductions in the air of the written alphabets (O, D, Z), the rest ones include separate conventional features of the written and printed letters. The dactyls' look is shown in Table 3 and it is also indicated whether the dactyl is static or dynamic.

Table 3 — Signs

A (А)	B (Б)	V (В)	G (Г)	D (Д)	Ye (Е)	Yo (Ё)	Zh (Ж)	Z (З)
								
static	static	static	static	dynamic	static	dynamic	static	static
I (И)	J (Й)	K (К)	L (Л)	M (М)	N (Н)	O (О)	P (П)	R (Р)
								
static	dynamic	dynamic	static	static	static	static	static	static
S (С)	T (Т)	U (У)	F (Ф)	X (Х)	C (Ц)	Ch (Ч)	Sh (Ш)	Shh (Щ)
								
static	static	static	static	static	dynamic	static	static	dynamic
`` (Ь)	Y` (Ы)	` (Ъ)	E` (Э)	Yu (Ю)	Ya (Я)			
								
dynamic	static	dynamic	static	static	static			

2.2 Dataset concept

For our project, we need a large dataset with all RSL dactyls, we use vectorized images to train our classifier model. For static dactyls, it is obvious that we need to collect images of each dactyl. For dynamic dactyls, this approach is not possible because they have many different positions in the showing process. My colleague Tsimafei Akulich studied hand movements while showing dynamic dactyls, based on which he divided them into two groups: dynamic dactyls without movements in the wrist joint (Z (З), C (Ц), Shh (Ш)) and dynamic dactyls with these movements (D (Д), Yo (Ё), J (Й), K (К), `` (Б), ` (В)). The essential difference between these two groups is that if you take a series consecutive picture while showing dactyls of the first group, the hand will look the same in them, for the second group the hand will rotate and have different positions in different pictures. Thus, for each dactyl of the first group there is one type of images, and the index fingertip trajectory of movement is taken into account when recognizing, for each dactyl of the second group there are several types of images from the sequence of hand positions when showing the dactyl, the order of appearance of these images is taken into account. As a result, the dataset contains 38 types of images for recognition of static and dynamic dactyls.

2.3 Train dataset creation

We created our own dataset [1], in the creation of which seven people participated. In addition to the original images, there are augmented images: each image was modified in four different ways (horizontal compression by 10%, horizontal stretching by 10%, vertical compression by 20%, vertical stretching by 20%). Thus, the train dataset contains 47320, whole dataset contains 62469 images. The model is tested on data that were not used during its training. An example of images from the test dataset is shown in Figure 5.

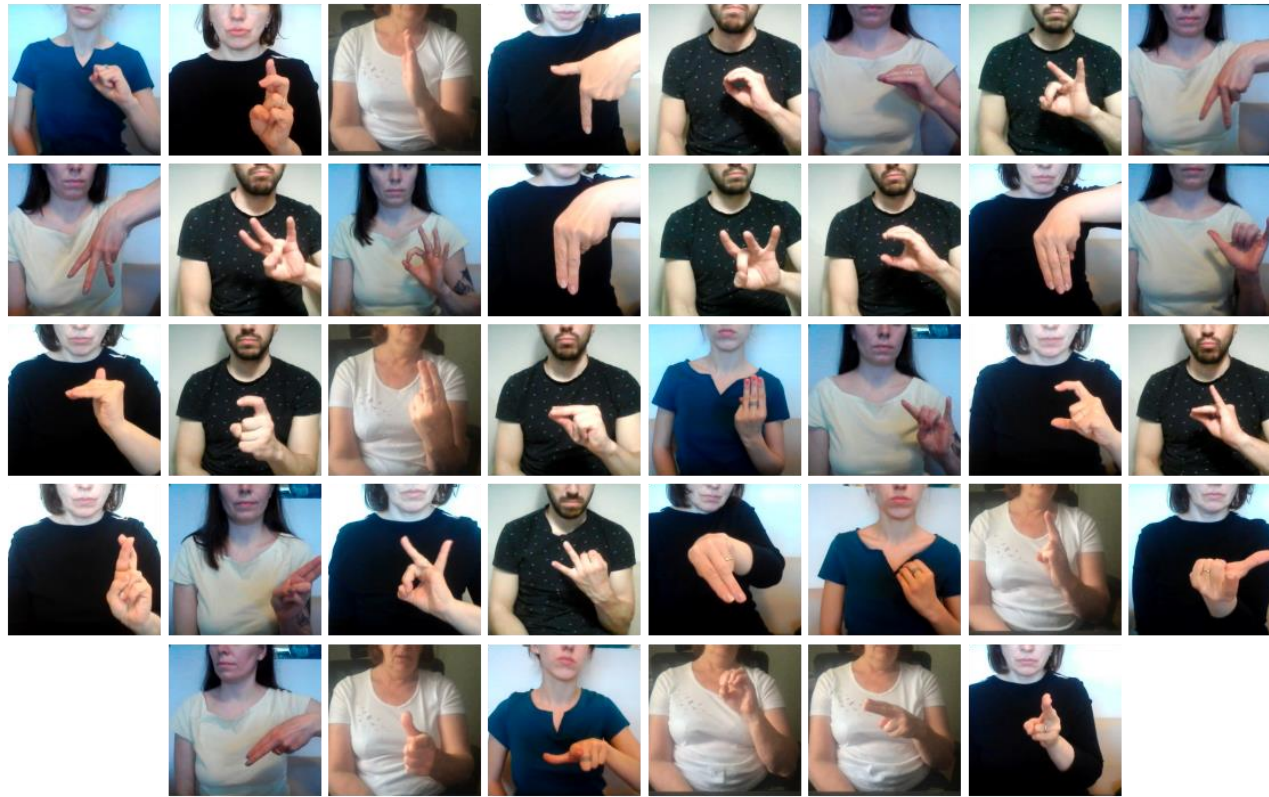


Figure 5 — RDA dataset examples

CHAPTER 3. PIPELINE

3.1 Hand pose estimation and landmarks

When training the model we use vectorized images. The first step is getting coordinates of hand points from the image using MediaPipe Hands [9], which consists of two models: Palm Detection Model and Hand Landmark Model. My colleague Tsimafei Akulich has written program code to convert video to images and obtain for images the coordinates of 21 points, each of which corresponds to an anatomical formation of the hand, using MediaPipe Hands. The obtained hand points are shown in Figure 6. Example of working shown in Figure 7.

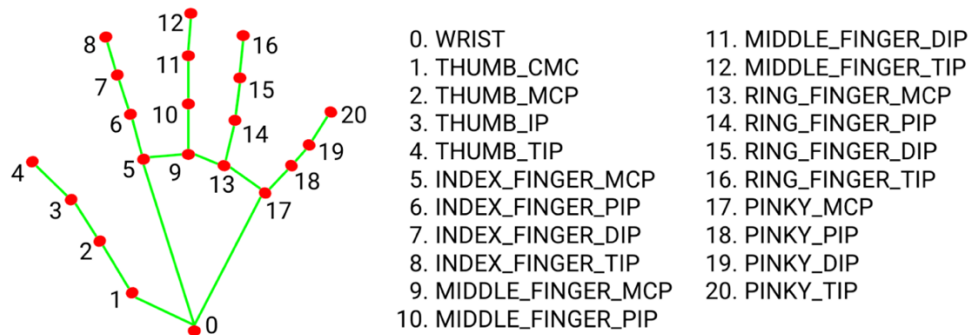


Figure 6 — Hand points defined by MediaPipe Hands

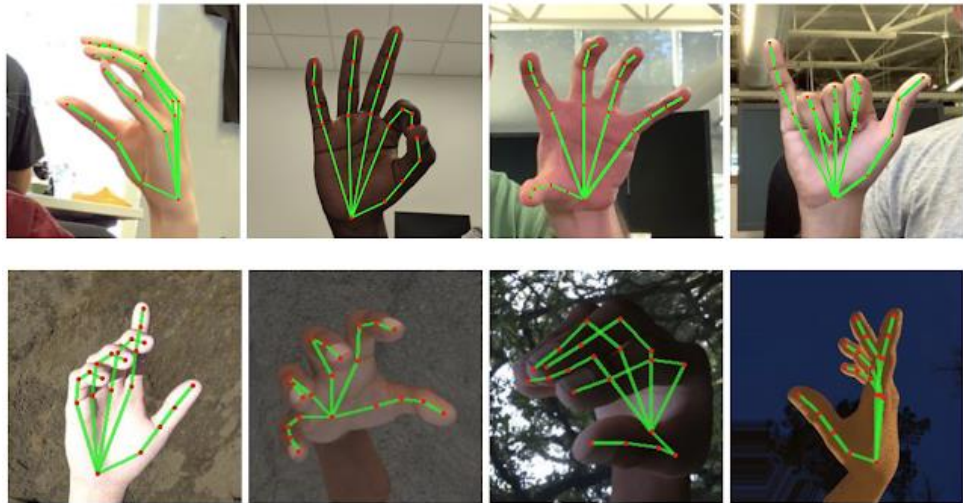


Figure 7 — An example of how MediaPipe Hands works

3.2 Feature generation

One of my tasks was to create an algorithm to vectorize the output data into features. As mentioned above, the output data are 21 points in three-dimensional space, that is, the coordinates of points on the x, y, z axes.

Some points have bones or almost fixed ligaments between them, that is, the distance between them does not change at all or changes insignificantly. Other points have a joint between them, it means that the distance between them varies with a movement. There are joints in the hand in which movements are possible along all the axes, and joints in which movements are performed along one axis, that is, points that change position independently of each other, and points that can change position only together and in one direction. These anatomical features were considered, and distances were measured between only those points where necessary, thus the distances between the following pairs of points are obtained:

- 0 and 2, 0 and 3, 0 and 4, 0 and 6, 0 and 7, 0 and 8, 0 and 10, 0 and 11, 0 and 12
0 and 14, 0 and 15, 0 and 16, 0 and 18, 0 and 19, 0 and 20;
- 3 and 6, 3 and 10, 3 and 14, 3 and 18;
- 4 and 8, 4 and 12, 4 and 16, 4 and 20;
- 6 and 10, 6 and 14, 6 and 18;
- 8 and 12, 8 and 16, 8 and 20;
- 10 and 14, 10 and 18;
- 12 and 16, 12 and 20;
- 14 and 18;
- 16 and 20.

Then we normalize these distances, so that it becomes irrelevant whether the hand is closer to the camera or farther away. In this way, we get 35 features, these features are provided at the Figure 8.

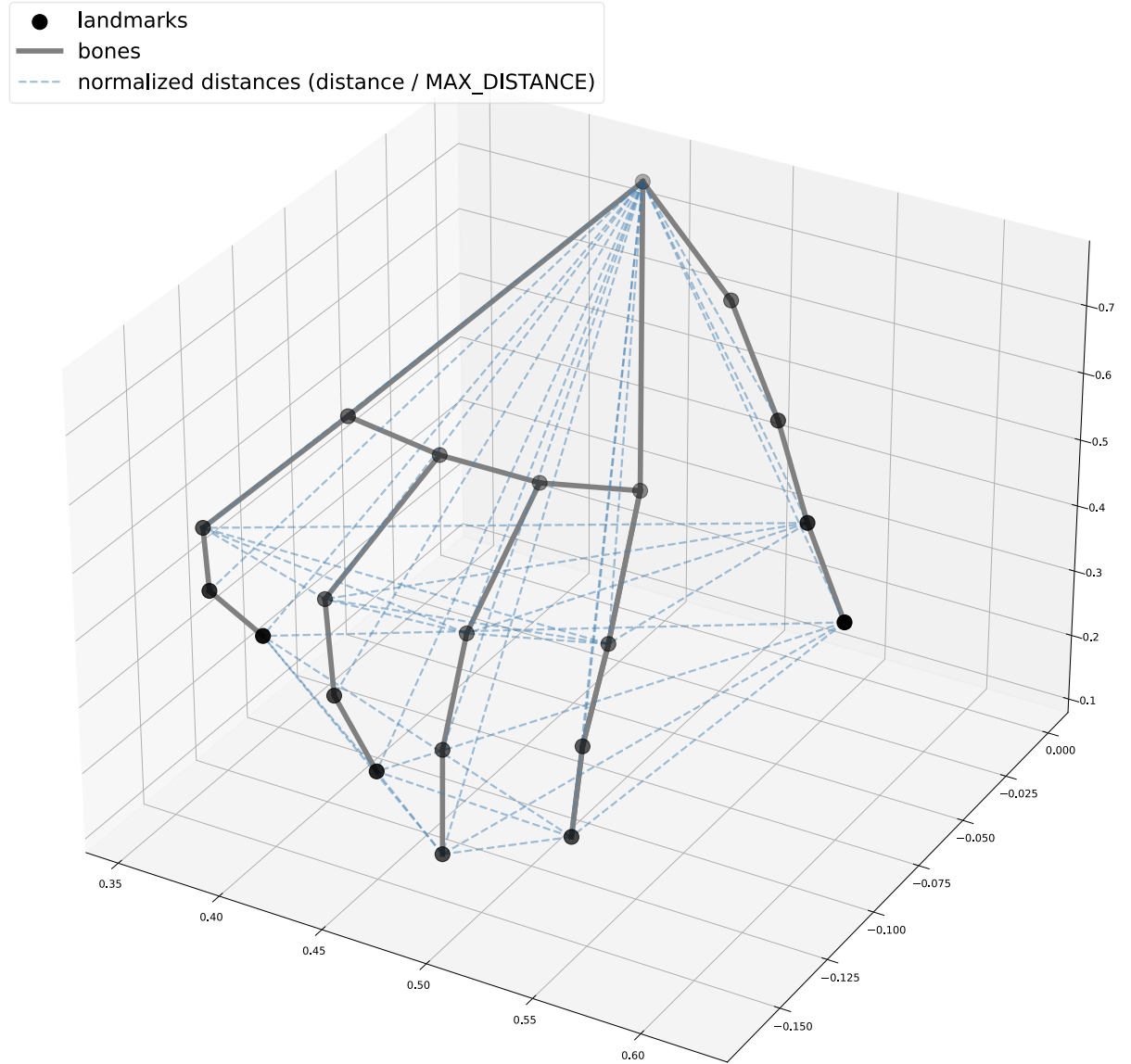


Figure 8 — Distances

The distances obtained are not sufficient, since there are dactyls in which the finger configuration is identical, but, for example, in one dactyl the hand is pointing upwards and in another downwards. I studied all the dactyls to find important differences between them and to create additional features that would be able to distinguish dactyls that are identical in finger configuration from each other.

Dactyls can be divided into several groups based on the degree of similarity:




1. Dactyl signs that do not become any other sign when the position of the upper

extremity is changed: A (A), B (Б), V (В), Zh (Ж), L (Л), M (М), N (Н), O (О), R (Р), S (С), U (У), F (Ф), Ch (Ч), Y` (Ы), E` (Э), Yu (Ю), Ya (Я).

2. Dactyl signs that become some other sign when the position of the upper extremity changes can be divided into subgroups in which the positions of the fingers are identical:





2.1. Dactyls G (Г), `` (Ђ), ` (Б) are shown in Table 4.

Table 4 — Similar dactyls G, `` , `

G (Г)	`` (Ђ)	` (Б)
		
static dactyl, index finger pointing down, hand facing palm toward the demonstrator.	dynamic dactyl, index finger pointing straight ahead, thumb moving from left to right, hand facing palm down.	dynamic dactyl, index finger pointing straight ahead, thumb moving from right to left, hand facing palm down.

2.2. Dactyls D (Д), K (К), P (П), C (Ц) are shown in Table 5.

Table 5 — Similar dactyls D, K, P, C

D (Д)	K (К)	P (П)	C (Ц)
			
dynamic dactyl, at the beginning of the movement the index and middle fingers are directed upwards, then the fingertips describe the contours of the capital letter «D», the trajectory is like a circle, the hand faces the back surface to the demonstrator.	dynamic dactyl, at the beginning of the movement, the index and middle fingers are directed upward, then the fingertips describe a quarter circle, the center of which is the wrist, the back surface of the hand is located laterally.	static dactyl, the index and middle fingers pointing down, palm surface of the hand facing the demonstrator.	dynamic dactyl, the index and middle fingers pointing downward, palm surface of the hand facing the demonstrator, the sign shifts downward.

2.3. Dactyls Ye (E), Yo (Ё) are shown in Table 6.

Table 6 — Similar dactyls Ye, Yo

Ye (E)



static dactyl, the back surface of the hand is located laterally.

Yo (Ё)



dynamic dactyl, the movement begins with the position of dactyl E, then the fingertips describe a quarter circle, the center of the circle is the wrist.

2.4. Dactyls Z (З), X (Х) are shown in Table 7.

Table 7 — Similar dactyls Z, X

Z (З)



dynamic dactyl, the back surface of the hand is lateral, the tip of the bent index finger describes the contours of the capital letter «Z».

X (Х)



static dactyl, the back surface of the hand is located laterally.

2.5. Dactyls I (И), J (Й) are shown in Table 8.

Table 8 — Similar dactyls I, J

I (И)



static dactyl, the ring finger and little finger pointing upward, the back of the hand facing the demonstrator.




J (Й)



dynamic dactyl, the ring finger and little finger are directed upwards, at the beginning of the movement the palm surface of the hand faces the demonstrator, then there is a rotation of the hand, and the hand takes the position of dactyl «I».

2.6. Dactyls T (T), Sh (III), Shh (III) are shown in Table 9.

Table 9 - Similar dactyls T, Sh, Shh

T (T)	Sh (III)	Shh (III)
		
static dactyl, the tips of the index, middle and ring fingers pointing downwards, palm facing the demonstrator.	static dactyl, with the tips of the index, middle, and ring fingers pointing upward and the palm facing the demonstrator.	dynamic dactyl, the tips of the index, middle, and ring fingers pointing up, palm facing the demonstrator, there is a downward movement.

Dynamic dactyl signs can be divided into several groups based on the presence or absence of a combination of unique finger placement and unique hand position:

1. Dynamic dactyl signs, that have combinations of unique finger placement and unique hand positions: D (Д), Yo (Ё), J (Й), K (К), C (Ц).
2. Dynamic dactyl signs, which differ from a similar dactylic sign only by the trajectory of movement:
 - 2.1. Z (З), similar dactyl — X (Х), the difference is that X is static, while Z is dynamic.
 - 2.2. Shh (III), similar dactyl — Sh (III), the difference is that Sh is static, while Shh is dynamic.
 - 2.3. `` (Б), ` (В): similar dactyls, both dactyls are dynamic, the difference is that the movement in `` is from left to right and in ` from right to left.

All the distinctive characteristics of dactyls were taken into account and six additional features were created; these features tell us the position of the hand in space. We take coordinates 0 and 17, the first one is displacement along the x-axis, the second one along the y-axis, and the third one along the z-axis.

We look on which axis the difference between the values of these two points is maximal, and on which axis the difference between the values of these two points is minimal, and on those axis we assign value 0, and to the remaining feature we assign

value 0.5. Then we see which of the points has a greater value on each of the axes, if the point has 0, then we make the value of the attribute negative, it helps to understand the direction of the brush rotation on each of the axes. We do the same for points 5 and 17. Thus, we vectorize the original images and obtained 41 features for each observation.

3.3 Model selection for dactyls classification

We need one more model to predict the class, the task is a multiclass classification. In my work I analyzed three models: Random Forest Classifier [12], C-Support Vector Classification [13] and LightGBM Classifier [14]. For each model the best hyperparameters were selected using GridSearchCV [15], it is an automated tool to find the best hyperparameters for a model by a given metric using crossvalidation. Learning time, prediction time, overall accuracy, and standard deviation were considered in selecting the best model, the best model is Random Forest Classifier.

The models used are shown below:

- RandomForestClassifier


```
{
'bootstrap': True, 'ccp_alpha': 0.0, 'class_weight': 'balanced', 'criterion':
'gini', 'max_depth': None, 'max_features': 'auto', 'max_leaf_nodes': None,
'max_samples': None, 'min_impurity_decrease': 0.0, 'min_samples_leaf': 3,
'min_samples_split': 2, 'min_weight_fraction_leaf': 0.0, 'n_estimators': 50,
'n_jobs': -1, 'oob_score': False, 'random_state': 42, 'verbose': 0, 'warm_start':
False
}
```
- SVC


```
{
'C': 1, 'break_ties': False, 'cache_size': 200, 'class_weight': 'balanced',
'coef0': 0.0, 'decision_function_shape': 'ovr', 'degree': 2, 'gamma': 'scale',
'kernel': 'poly', 'max_iter': -1, 'probability': True, 'random_state':
42, 'shrinking': True, 'tol': 0.001, 'verbose': False
}
```
- LGBMClassifier


```
{
'boosting_type': 'goss', 'class_weight': 'balanced', 'colsample_bytree': 1.0,
'importance_type': 'split', 'learning_rate': 0.1, 'max_depth': -1,
'min_child_samples': 20, 'min_child_weight': 0.001, 'min_split_gain': 0.0,
'n_estimators': 100, 'n_jobs': -1, 'num_leaves': 100, 'objective': None,
'random_state': 42, 'reg_alpha': 0.0, 'reg_lambda': 0.0, 'silent': 'warn',
'subsample': 1.0, 'subsample_for_bin': 200000, 'subsample_freq': 0
}
```

Dactyl prediction overall accuracies on cross validation and test sample, fitting and predicting times are shown in Table 10, descriptive statistics are shown in Table 11, accuracies of each model for each dactyl are shown in Table 12. The best model is Random Forest Classifier, it has high accuracy on cross validation, high accuracy on the test sample, the smallest standard deviation, that is the most stable classifies different dactyls, the prediction time is the smallest.

Table 10 — Model comparison

	Accuracy on cross-val	Accuracy on test	Fitting time (sec.)	Predicting time (sec.)
SVC	94.58%	95.33%	14.67	8.73
RandomForestClassifier	93.72%	95.06%	4.86	0.35
LGBMClassifier	90.91%	94.63%	33.55	4.39

Table 11 — Models accuracies descriptive statistics

	Random Forest Classifier	SVC	LGBM Classifier
count	33.00	33.00	33.00
mean	95.06%	95.33%	94.63%
std	7.94%	10.46%	7.89%
min	71.40%	42.40%	67.80%
25%	94.80%	95.80%	91.60%
50%	98.40%	98.40%	98.60%
75%	100.00%	100.00%	100.00%
max	100.00%	100.00%	100.00%

Table 12 — Accuracy results

Letter of RDA	Transliteration	Random Forest Classifier	SVC	LGBM Classifier
A	A	100.0%	100.0%	100.0%
Б	B	87.4%	89.2%	91.6%
В	V	100.0%	100.0%	100.0%
Г	G	98.4%	98.2%	98.4%
Д	D	80.69%	85.17%	85.48%
Е	Ye	98.0%	85.2%	98.4%
Ё	Yo	97.6%	91.2%	92.9%
Ж	Zh	100.0%	98.2%	99.2%
З	Z	97.6%	97.2%	99.0%
И	I	71.4%	42.4%	67.8%
Й	J	82.5%	97.9%	88.4%
К	K	71.4%	87.7%	75.7%
Л	L	94.4%	95.8%	93.8%
М	M	100.0%	96.4%	98.6%
Н	N	100.0%	100.0%	100.0%
О	O	100.0%	100.0%	100.0%
П	P	90.8%	99.8%	85.6%
Р	R	94.8%	100.0%	99.8%
С	S	100.0%	100.0%	99.8%
Т	T	88.8%	96.6%	87.4%
У	U	100.0%	100.0%	100.0%
Ф	F	100.0%	100.0%	100.0%
Х	X	97.6%	97.2%	99.0%
Ц	C	100.0%	100.0%	100.0%
Ч	Ch	100.0%	100.0%	98.6%
Ш	Sh	99.8%	100.0%	100.0%
Щ	Shh	99.8%	100.0%	100.0%
Ъ	``	95.27%	95.2%	95.6%
Ы	Y`	97.2%	98.4%	87.8%
Ь	`	95.27%	95.2%	95.6%
Э	E`	100.0%	100.0%	100.0%
Ю	Yu	99.2%	99.0%	100.0%
Я	Ya	99.2%	99.8%	84.2%
Overall		95.06%	95.33%	94.63%

3.4 Results comparison and selection of the best model

My colleague Tsimafei Akulich analyzed 3 models: Logistic Regression [16], Decision Tree Classifier [17] and Multi-layer Perceptron with two hidden layers. Learning time, prediction time, overall accuracy, and standard deviation were considered in selecting the best model, the best model is Logistic Regression.

We compared the results obtained by Random Forest Classifier and Logistic Regression; descriptive statistics of the obtained accuracies are presented in Table 13. The overall accuracy of Linear Regression is higher, the standard deviation is lower, Logistic Regression was chosen as the best model and will be used in the application.

Table 13 — Comparison results

	Logistic Regression	Random Forest Classifier
count	33.00	33.00
mean	95.75%	95.06%
std	6.67%	7.94%
min	75.00%	71.40%
25%	95.40%	94.80%
50%	98.00%	98.40%
75%	100.00%	100.00%
max	100.00%	100.00%

The test sample consists of original images and augmented images, it is necessary to check the quality of the model separately on each part of the sample. The results of the test are presented in Table 14. According to the descriptive statistics, we can see that the overall accuracy for the original images and the overall accuracy for the augmented images are almost the same, and the standard deviations are almost the same. It can be concluded that the model classifies well not only the original images, but also the compressed or stretched images, thus, it can be assumed that the model will work equally well on the new test data.

Table 14 — Models accuracies descriptive statistics

	Original images	Images with augmentation
count	33	33
mean	95.96%	94.84%
std	8.04%	8.03%
min	72.00%	70.00%
25%	97.67%	94.25%
50%	100.00%	98.00%
75%	100.00%	100.00%
max	100.00%	100.00%

CHAPTER 4. TESTING THE MODELS UNDER DIFFERENT CONDITIONS

It is necessary to investigate the performance of the models under different conditions, such as the distance between the user and the camera, the level of illumination, the rotation of the user's hand, and the color of the background.

4.1 Distance

The possibility of hand detection and recognition of dactyls from different distances was investigated, the following distances in centimeters (distance between the webcam and the hand) were investigated: 400, 300, 250, 200, 150, 100, 80, 60, 50, 40, 30, 20. There are 100 observations for each distance.

The test results are shown in Table 15, examples of images obtained from different distances are shown on Figure 9.

Table 15 - Distance sensitivity

Distance (cm)	Hand palm detection accuracy	Dactyl recognition accuracy
400	0%	0%
300	18%	0%
250	32%	9%
200	98%	89%
150	99%	99%
100	100%	96%
80	100%	100%
60	100%	100%
50	100%	98%
40	100%	100%
30	100%	100%
20	100%	95%

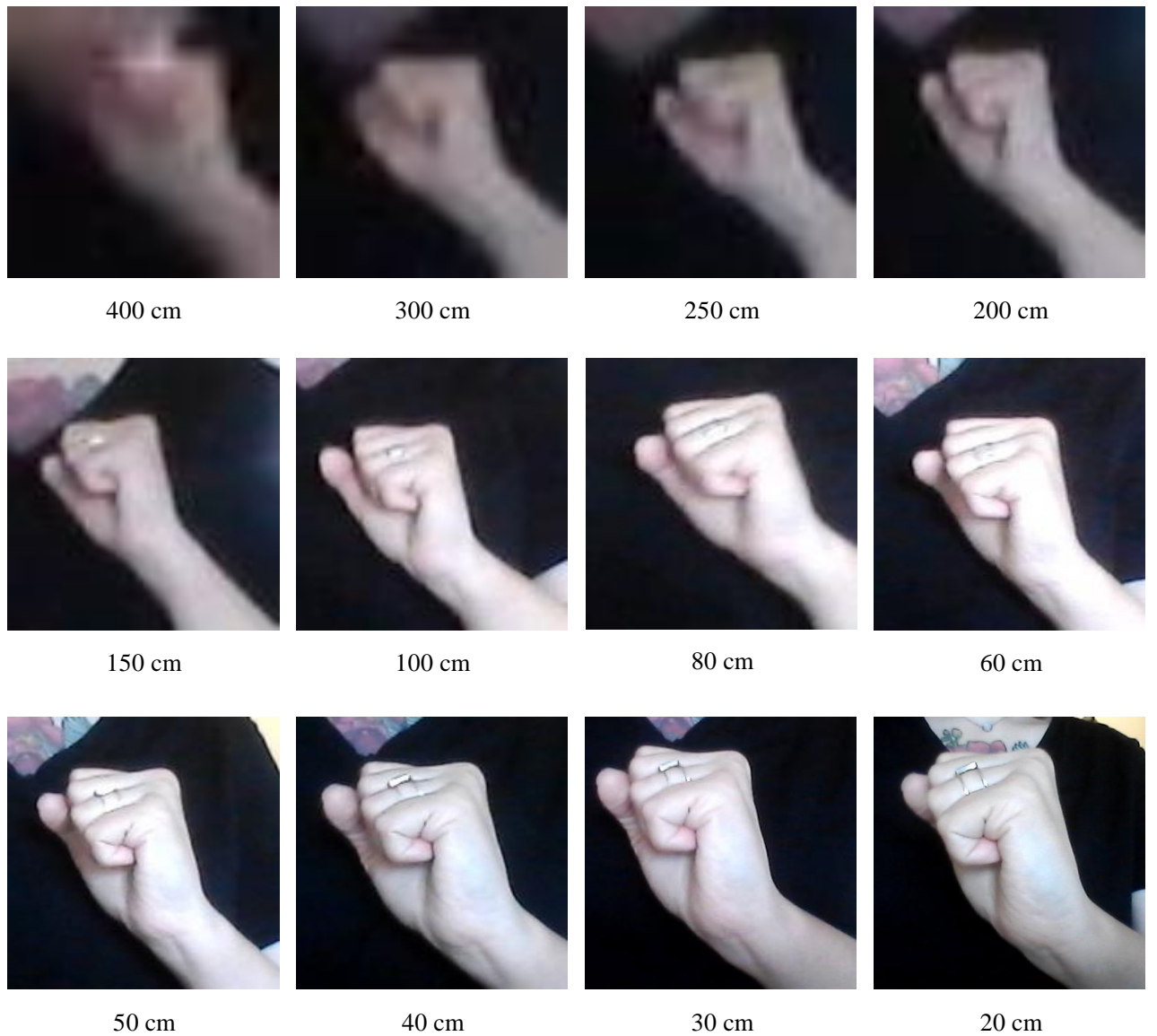


Figure 9 — Distances examples

We can say that it is acceptable to use the models when the distance between the user's hand and the webcam is 20 to 200 centimeters, the optimal distance is 30 to 150 centimeters. According to the results, we can say that at larger distances between the user's hand and the camera, there are both difficulties in hand detection and difficulties in correctly classifying dactyls.

4.2 Illumination level

The possibility of hand detection and recognition of dactyls at different illumination levels was investigated, the following illumination levels in lux were

investigated: 3, 10, 15, 30, 50, 80, 100, 300, 500, 650, 800, 1000. There are 100 observations for each level of illumination. The illumination level is measured in lux, the higher the value, the brighter the place where the user is.

The test results are shown in Table 16, examples of images obtained from different distances are shown on Figure 10.

Table 16 - Illumination sensitivity

Illumination level (lux)	Hand palm detection accuracy	Dactyl recognition accuracy
3	78%	78%
10	100%	100%
15	100%	100%
30	100%	100%
50	100%	100%
80	99%	100%
100	100%	100%
300	98%	100%
500	79%	79%
650	68%	68%
800	5%	5%
1000	0%	0%

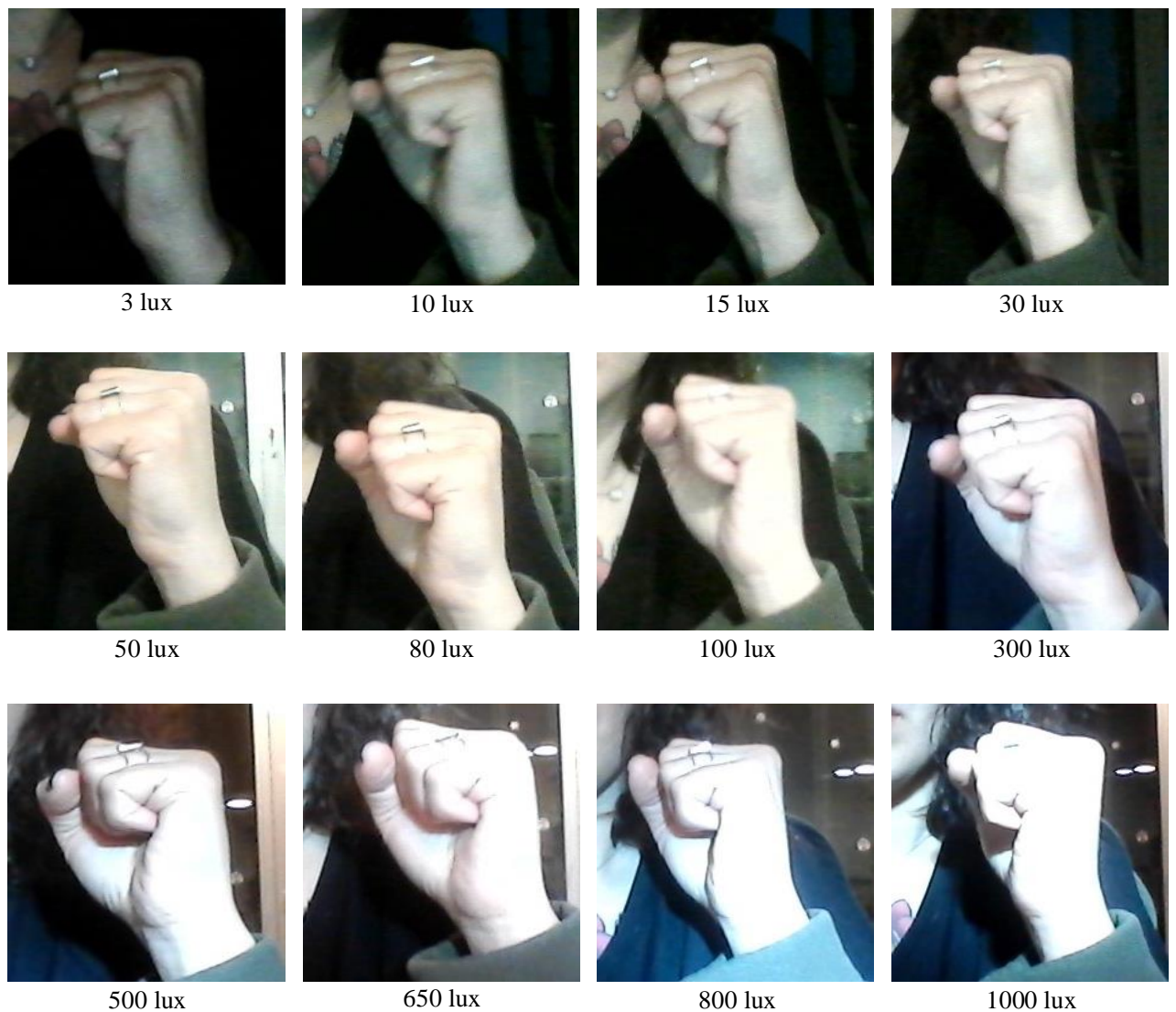


Figure 10 — Illumination examples

We can say that it is acceptable to use the models when the light level is between 3 and 500 lux, that is, from a low level of light, such as light from a night light to a medium level of light, such as daylight. The optimal light level is 10 to 300 lux. When very bright light is directed at the user, the image becomes glitzy. Based on the results obtained, it can be said that at high light levels it is difficult to detect the hand.

4.3 Rotation

My colleague Tsimafei Akulich investigated the possibility of recognizing dactyls at different turns of the hand was investigated, turns in a circle with a step of

15 degrees were investigated. The following accuracy was obtained:

- 0 degrees, 100.0%
- 15 degrees, 96.4%
- 30 degrees, 76.0%
- 45 degrees, 27.2%
- 60 degrees, 3.6%
- from 75 to 180 degrees and from -105 to -165 degrees, 0%
- -90 degrees, 1.6%
- -75 degrees, 6.4%
- -60 degrees, 68.4%
- -45 degrees, 95.6%
- -30 degrees, 100.0%
- -15 degrees, 100.0%

There are 250 observations for each turn. We can say that turning the hand by 30 degrees to either side of the correct hand position when showing a dactyl is acceptable, it is optimal not to change the position by more than 15 degrees to either side. According to the results, we can say that if the position of the hand differs significantly from the correct position, there are difficulties in recognizing dactyls. It is important to note that there are correct ways to show each dactyl, so low accuracy with too much change in hand position is more of a user error than a model, the user should be able to show the dactyl correctly.

4.4 Color of clothes (background)

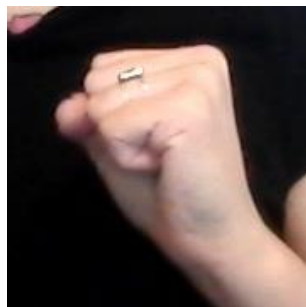
The possibility of hand detection and recognition of dactyls in different backgrounds was investigated, the following background colors were investigated: black, beige, bright multicolored. There are 100 observations for each background.

The test results are shown in Table 17, examples of images obtained with

different background colors are shown in Figure 11.

Table 17 — Color sensitivity

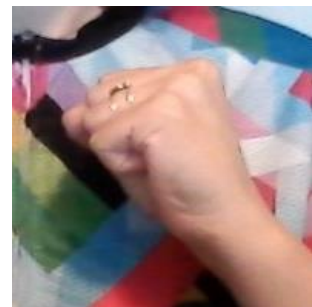
Color	Hand palm detection accuracy	Dactyl recognition accuracy
black	100%	99%
beige	100%	100%
bright multicolored	100%	80%



black



beige



bright multicolored

Figure 11 — Color examples

We can say that it is better to use the models while wearing monochrome clothing. This rule is relevant not only for working with the application, but also sign language interpreters should wear monochrome clothes while working, as it is more difficult for the eye to recognize a dactyl on a multicolored background.

CHAPTER 5. APPLICATION

5.1 Application description

My colleague Tsimafei Akulich developed the architecture for dactyl education application. The application is presented as a web service written in python [18] programming language and FastAPI [19] as web framework. The application has three modes: Simple detection mode, Correct dactyl mode, Word guessing mode. Also, my colleague developed SDK for faster development of new modes.

A sequence diagram of the application is shown in the Figure 12.

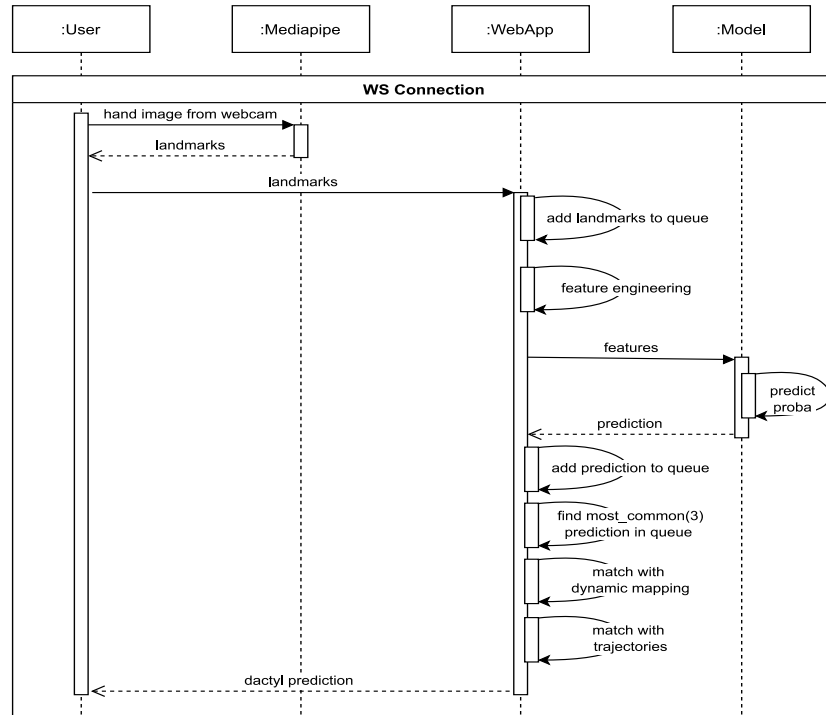


Figure 12 — Sequence diagram

5.2 Application flow

Video from the webcam is divided into frames, 30 frames are recorded every second, the frames are stored for one second, and the resulting images are sent to

MediaPipe Hands. MediaPipe Hands returns for each image the coordinates of 21 points in three-dimensional space, each point representing an anatomical object of the hand. The coordinates are sent to the web application, where 41 features are generated for each set of coordinates using the previously described algorithm. In addition, the coordinates of the index fingertip are recorded for each image, this is needed to track the movement trajectory. The web application then sends the features to the Logistic Regression model, the model returns the class prediction and the probability of belonging to the class, if the probability is greater than 0.9, then this prediction is placed in the queue of possible predictions. The web application then calculates the trajectory of the index fingertip. From all the predicted classes the three most frequent ones are taken, the application compares the result of the obtained trajectory of movements with the possible ones, if a match was obtained, one of the dynamic dactyls is predicted, otherwise one of the static ones.

5.3 Application operation modes

5.3.1 Simple detection and classification mode

My colleague made the first mode of the application - simple detection and classification of dactyls. In this mode, the app simply displays the name of the dactyl in the output field if the finger configuration and movement trajectory correspond to one of the dactyls. If no hand is shown, or if a finger configuration is shown that is not a dactyl, the output field is left blank. Figure 13 shows this mode.

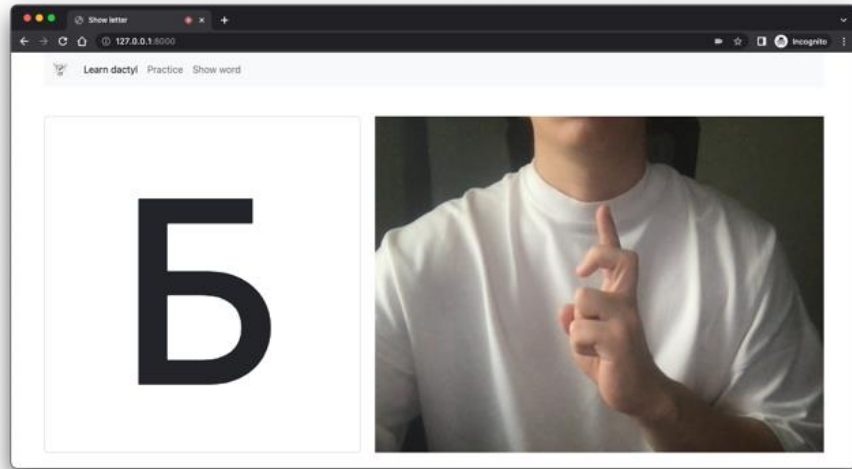


Figure 13 — Dactyl B

5.3.2 Correct dactyl mode

My colleague created the second mode of the application, which is a mode for checking the correctness of the dactyl shown. In this mode, a randomly selected letter appears in the output field, the user needs to show the corresponding dactyl. There is a possibility to get a hint - a picture of a dactyl, if the user can't remember a dactyl. Figure 14 shows this mode.

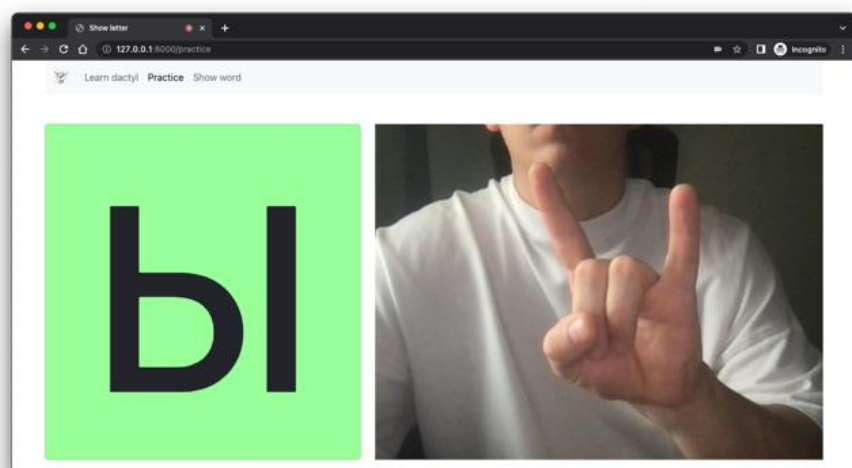


Figure 14 — User shows correct dactyl

5.3.3 Word guessing mode

The third mode of the application is the word guessing mode. A dataset of 1000 popular Russian words [2] was loaded, each time a word from the dataset is randomly chosen. The API [20] searches for pictures corresponding to this word, the first six are chosen, the web application shows the user these pictures and the length of the word, each letter is an empty square, Figure 15. The user guesses the puzzled word from the pictures and the length of the word and shows it letter by letter, Figure 16. After the user enters all the letters, the web application moves on to the next word.

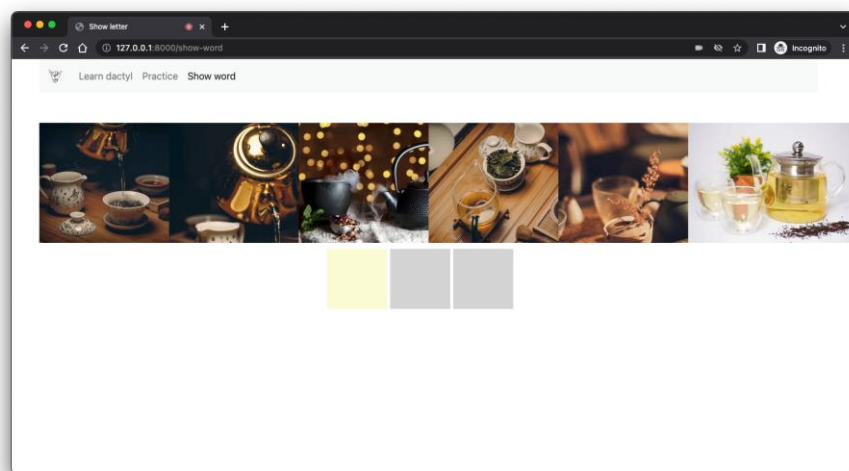


Figure 15 — Random word showing using images

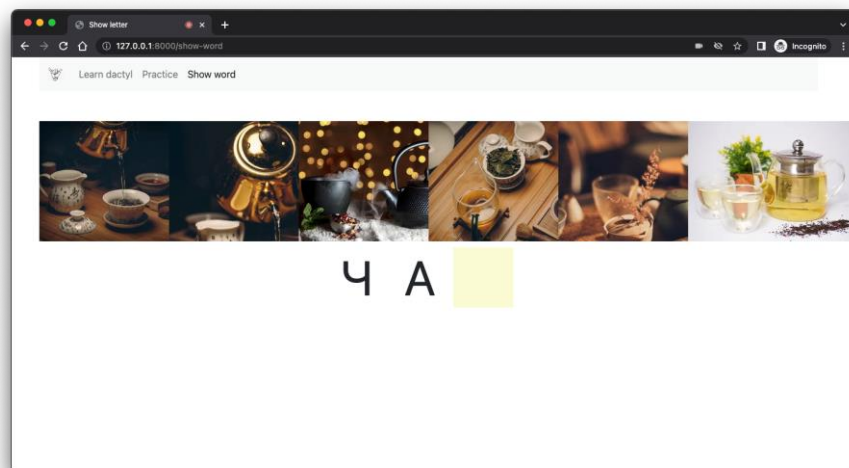


Figure 16 — User inputs word letter by letter using dactyls

CONCLUSION

The goal of our project was to implement recognition of all the dactyls of RSA and obtain a higher overall accuracy with a smaller standard deviation. Since there are dynamic dactyls, this is only possible with recognition from a video. We have successfully achieved our goal and made the application to make our project complete. Table 18 has descriptive statistics for the recognition accuracies of the dactyls, the first column shows the results for 26 dactyls (24 of which are static and 2 dynamics, but presented as static) using the QuadroConvPoolNet model, which previously gave the best result [2]. The second column shows the results for these 26 dactyls using the current approach (MediaPipe Hands, Logistic Regression and author's algorithms), the third column shows the results for 7 remaining dactyls using the current approach.

Table 18 — Descriptive statistics of dactyl prediction accuracies

	QuadroConvPoolNet, static dactyls and two dynamic dactyls represented as static	Current approach, static dactyls and two dynamic dactyls	Current approach, other dynamic dactyls
count	26	26	7
mean	77.53%	96.08%	93.71%
std	8.99%	6.97%	5.94%
min	61.00%	75.00%	81.00%
25%	69.00%	96.00%	94.50%
50%	80.00%	99.50%	95.00%
75%	84.75%	100.00%	95.5%
max	93.00%	100.00%	100%

We got the following results: average accuracy increased by 18.55%, standard deviation decreased by 2.02%, minimum accuracy increased by 14%, maximum accuracy increased by 7% to 100%. We believe that we achieved our goal and got great results. In the future, it is possible to implement dactyl recognition for other sign languages and to implement word recognition for sign languages using our approach.

REFERENCES

1. I. Makarov, N. Veldyaykin, M. Chertkov, and A. Pokoev. 2019. “American and Russian Sign Language Dactyl Recognition”, in: Proceedings of the 12th ACM Conference on Pervasive Technologies Related to Assistive Environments (PETRA’19). NY: ACM, 2019. P. 204-210. // <https://dl.acm.org/doi/10.1145/3316782.3316786>
2. I. Makarov, N. Veldyaykin, M. Chertkov, and A. Pokoev. 2019. “Russian Sign Language Dactyl Recognition”, in: 2019 42nd International Conference on Telecommunications and Signal Processing (TSP). NY: IEEE, 2019. P. 726-729. // <https://ieeexplore.ieee.org/document/8768868>
3. I. Makarov, N. Veldyaykin, M. Chertkov, and A. Pokoev. 2019. “American and Russian Sign Language Dactyl Recognition and Text2Sign Translation in Analysis of Images”, in: Analysis of Images, Social Networks and Texts. 8th International Conference AIST 2019. Springer, 2019. P. 309-320. // https://dl.acm.org/doi/abs/10.1007/978-3-030-37334-4_28
4. Murat Taskiran, Mehmet Killioglu, and Nihan Kahraman. 2018. “A Real-Time System for Recognition of American Sign Language by using Deep Learning”, in: 2018 41st International Conference on Telecommunications and Signal Processing (TSP). IEEE, IEEE, New York, NY, USA, 2018. P. 1–5. // <https://ieeexplore.ieee.org/document/8441304>
5. Debasrita Chakraborty, Deepankar Garg, Ashish Ghosh, and Jonathan H. Chan. 2018. “Trigger Detection System for American Sign Language using Deep Convolutional Neural Networks”, in: Proceedings of the 10th International Conference on Advances in Information Technology. ACM, ACM DL, New York, NY, USA, 2018. P. 1-6. // <https://dl.acm.org/doi/10.1145/3291280.3291783>
6. Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. “Gradient based learning applied to document recognition”. Proc. IEEE 86, 11 (1998). P. 2278– 2324. // <https://ieeexplore.ieee.org/document/726791>
7. G. Rao, K. Syamala, P. Kishore, and A. Sastry. 2018. “Deep convolutional neural networks for sign language recognition”, in: Signal Processing and Communication Engineering Systems (SPACES), 2018 Conference on. IEEE, IEEE, New York, NY, USA, P. 194–197. // <https://ieeexplore.ieee.org/abstract/document/8316344>
8. Transliteration of Russian alphabet, GOST-7.79-2000 (system B) // <http://transliteration.ru/gost-7-79-2000>
9. MediaPipe Hands // <https://google.github.io/mediapipe/solutions/hands>
10. MediaPipe Palm Detection Model // <https://google.github.io/mediapipe/solutions/hands#palm-detection-model>

11. MediaPipe Hand Landmark Model // <https://google.github.io/mediapipe/solutions/hands#hand-landmark-model>
12. Random Forest Classifier // <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html>
13. C-Support Vector Classification // <https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html>
14. LightGBM Classifier // <https://lightgbm.readthedocs.io/en/latest/pythonapi/lightgbm.LGBMClassifier.html>
15. Grid Search CV // https://scikit-learn.org/stable/modules/grid_search.html
16. Logistic Regression // https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html
17. Decision Tree Classifier // <https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html>
18. Python Programming Language // <https://www.python.org/>
19. FastAPI // <https://fastapi.tiangolo.com/>
20. Pixels API // <https://api.pexels.com>

APPLICATIONS

1. Dataset // <https://github.com/timaakulich/diploma-applications/tree/master/dataset>
2. Popular Russian words // https://github.com/timaakulich/diploma-applications/blob/master/popular_russian.txt
3. Web application source code // <https://github.com/timaakulich/diploma>