Subsystem for Simple Dynamic Gesture Recognition Using 3DCNNLSTM

Mikhail Artemov, Vyacheslav Voronov, Lilia Voronova, Artem Goncharenko, Vasiliy Usachev
Moscow Technical University of Communications and Informatics
Moscow, Russian Federation
artemov_mikle@mail.ru, vorvi@mail.ru, voronova.lilia@ya.ru, artem@agsys.ru, usvas.01@gmail.com

***Abstract*— The article describes the subsystem of the intellectual information-communication system developed at MTUCI. The analysis of the latest developments and research in the field of sign language recognition systems has been carried out. The net architecture and the training method have been considered for 3DCNNLSTM model that will be used to recognize simple dynamic gestures. A training data set, initial conditions, experiments performed and the use of learning outcomes are described.**

I. INTRODUCTION

The program "Digital Economy of the Russian Federation", adopted in 2017, identifies priority areas for the development of science, technology and technology for the country.

In the Moscow Technical University of Communications and Informatics (MTUCI) there is a system of internal grants for scientifically-research work (SRW). As part of this system, in 2018, the department «Intelligent Systems in Management and Automation» conducted research work on the project «Developing an Intelligent Information and Communication System for Social Accessibility for People with Disabilities Based on Machine Learning Methods», scientific adviser – Candidate of Technical Sciences Vyacheslav I.Voronov [1], [2].

The object of SRW are algorithms, technologies and means of converting information between natural and sign languages based on machine learning methods [3], [4].

The goal of the project: the development of a software package based on machine learning methods that provide the functionality of the communication system of social accessibility for people with hearing impairments.

In the course of the work, research was carried out on models and methods of data mining to transform information from sign language into colloquial, designing tools for communication of hearing impaired people, including fingerprint recognition, gestures, gestures and emotions [5], [6].

As part of SRW, a large number of experiments on the recognition of gestures with the help of trained neural networks were conducted. The obtained results give basis for cautious optimism in connection with the active development of convolutional and recurrent neural networks and other data mining methods used for pattern recognition [7], [8], [9].

The obtained results were introduced into the educational process at MTUCI in the form of a laboratory workshop and methodological guide for the following disciplines: «Data mining methods» in the direction 15.04.04 – Automation of technological processes and production, the program «Artificial Intelligence Systems in IIoT» and «Machine learning. Technical systems, which can being trained» in the direction 27.04.04 – Management in technical systems, the program «Intelligent data analysis in technical systems».

The authors of the article present the results of the development of one of the directions of this research, a subsystem including a neural network that will perform the recognition of actions and simple dynamic gestures of people.

During the development of the «Surdotelephone» project, a simple convolutional neural network [1], [3], developed earlier and recognizing static gestures, ceased to meet the requirements and the 3DCNNLSTM (3-dimensional convolutional neural network with long short-term memory) neural network architecture was developed to replace it, consisting of three-dimensional convolutional layers, an LSTM cell and fully connected layers, the task of which is to recognize the dynamics of simple gestures and actions [10], [11]. Compared with two-dimensional convolutional layers, three-dimensional convolutional layers allow for convolution in the time domain, which is of critical importance in tasks related to movement, and the LSTM layer allows long-term memorization, making it possible to build complex temporal dependencies between the features derived from convolutional layers.

The 3DCNNLSTM model is based on a recurrent layer that receives information not only from the previous layer, but also from itself as a result of the previous data pass [12], which allows the model to recognize the dynamics in the input data sequence.

It is assumed that the sequence of images supplied to the input of the network contains a person showing hand gestures on a plain background. But this situation is idealized because the video may contain foreign objects and not have a uniform tone, besides it may have a different color palette, the person showing the gesture may be at different distances from the video camera, not be in focus, and the lighting in the frame may be far from perfect.

The presence of all this information in the frame should not affect the operation of the recurrent network, the network must accept a set of attributes that determine the person and what he shows on each frame. In addition, the extra information at the input affects the size of the data describing the state of the network at a specific point in time, the larger the size of the

data at the input, the more memory is required to store the network state, which also increases the time for calculations.

Therefore, a set of images is fed to the input of the convolutional network. The images are converted into a vector of pixels, which comes along the first convolutional layer of the network, from which a set of features (feature maps) is formed, which goes to the next convolutional layer and forms a new feature set, etc. As a result, at the output of the convolutional network, feature maps are formed, which enter the recurrent network.

To reduce the size of feature maps, subsampling layers are added to the convolutional network, which makes it possible to reduce the size of the inputs of the recurrent network, as well as the amount of memory used to store the network state.

The output of the recurrent network is a fully connected network, which, based on the results of calculations of the recurrent block, predicts the class to which the sequence of frames at the input belongs.

This article describes the model 3DCNNLSTM architecture, its learning process, as well as the training data set and metrics of the trained network at different eras.

II. Related works

To recognize static gestures used convolutional neural networks that can be trained on datasets consisting of images of the same color [13], and also trained on specially augmented datasets, for example, by constructing a virtual 3D hand model [14]. A slightly different approach relies on the use of Region-based CNN, which localizes the gesture of the hand and recognizes it [15].

A skeletal approach to the recognition of 3D hand gestures was proposed in [16], it is based on a deep learning model, where the sequences of the skeletal joints of the hand go to parallel convolutions. This model achieves 91.28% classification accuracy for 14 gestures and 84.35% for 28 gestures using the DHG dataset compiled using an Intel RealSense depth camera.

Existing developments on the recognition of dynamic gestures using neural networks can be divided into three groups according to the principle of working with the time dimension, as suggested in the article [17]:

1) Projects using 3D filters in the CNN convolutional layer. In this case, 3D convolution and 3D pooling capture the distinguishing features in the spatial and temporal dimension during the processing of each structure, as shown in Fig. 1, a.

2) Projects that use motion functions, such as dense 2D optical flow maps, which are pre-computed and fed to the input of the neural network, as shown in Fig. 1, b. Signs of motion calculated in this way can be used in the neural network as additional channels or enter the next neural network (proceeding further along with other signs).

3) Projects that combine 2D (or 3D) CNN, applied individually to each frame (or to a set of frames), simulating a time sequence, as shown in Fig. 1, c. One of the neural networks in the most popular of such tasks is the recurrent neural network (RNN), because it takes into account data that changes over time, using recurrent connections in hidden layers.

Among the works that use 3D convolution and 3D-pooling in CNN can be distinguished [18], [19]. In [18], an approach was proposed for teaching spatial-temporal features using deep three-dimensional convolutional networks (3D ConvNets), trained on a large amount of video-controlled data sets. This article summarizes three findings: 3D ConvNets is more suitable for the space-time learning of features in comparison with 2D ConvNets; the same number of convolutional kernels on all layers is more productive 3D ConvNets architecture than the architecture with heterogeneous sizes of cores on convolutional layers; The C3D (Convolutional 3D) architecture with a simple linear classifier is superior to modern methods and has 52.8% accuracy on the UCF101 data set in 10 dimensions and is effective in computing due to the rapid network prediction of ConvNets. Such a network is conceptually simple and easy to learn and use.

Another paper [19] proposes the alternation of channels of depth and intensity of an image for constructing normal space-time values, which will be used to train two neural subnetsTo work with networks, the VIVA dataset is used, consisting of 885 video recordings describing the intensity and depth of 19 hand gestures reproduced by eight people. All videos from the dataset were recorded using the Microsoft Kinect device under different lighting conditions and have a resolution of 15×250 pixels. Network prediction accuracy is 77.5%.

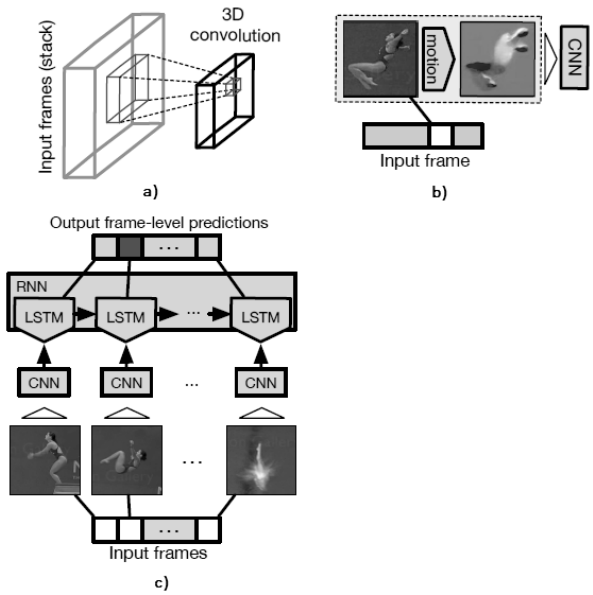


Fig. 1. Neural network groups a) - 3D convolution; b) - calculation of motion function; c) sequential simulation using LSTM [17]

The second group of networks includes [20], where a semi-learning approach using a deep learning network with slow merging, as well as temporal and spatial convolutions, including an autoencoder with a loss function for classification, being trained in parallel is proposed.

In the article [21] three views of deep sequences are proposed, respectively: dinamic depth images (DDI), dinamic depth normal images (DDNI), dinamic depth motion normal images (DDMNI).

The described dynamic images are made up of a sequence of depth maps, using ranking in space (position) and time (motion) directions at different levels to effectively capture the space-time information. These views allow the flexibility to customize the ConvNets model to classify deep sequences without incorporating large parameters into the training. Convolution networks developed using the proposed approach achieve 55.7% classification accuracy.

The third group of networks includes an alternative approach to gesture recognition [10]. It represents a model consisting of: a three-dimensional convolutional network (3D-CNN), a recurrent network (LSTM) and softmax-layers, which is trained on a very large annotated data set consisting of hand gestures video recordings.

For training the neural network uses a large dataset of short, tightly named videos, the image on which is as close as possible to the conditions of the real world. The total number of entries was 150,000 entries for 25 classes of gestures, divided in a ratio of 8:1:1 for training, validation and testing. The dataset also contains 2 classes of no gestures so that the network can distinguish gestures from simple hand movements. This neural network is able to provide video processing speed of 18fps and 87% accuracy when checking.

The same group of networks includes [11], where the Long Short-Term Memory Recurrent Neural Network (CNNLSTM) is proposed for recognizing dynamic gestures. The model contains two convolutional layers, a dense layer, a recurrent LSTM layer and a softmax layer at the output. The network input receives the result of a bitwise AND operation of every three frames of the video. The trained network demonstrates the following results: recognition accuracy is 91.67% ± 1.13%, accuracy is 92.25% ± 1.02%, memory is 91.67% ± 1.13%, F1-measurement is 91.63% ± 1.15%.

In [22], an extensive assessment of the task of dynamic recognition of hand gestures in the RNN study for visual study of the sequence is carried out. In particular, in order to use the powerful synthesis potential of pre-trained convolutional neural networks (CNN), where a new and effective approach is proposed, PreRNN, to make pre-trained CNN repetitive by converting convolutional layers or fully connected layers into repeating layers. These experiments show that PreRNN is superior to traditional RNN and achieves state-of-the-art results, suggesting that PreRNN is more suitable for studying visual sequences.

III. Architecture of 3DCNNLSTM

The proposed architecture for recognizing dynamic gestures consists of a sequence of convolutional layers, a layer of long-term short-term memory and a sequence of fully connected layers at the output of the neural network, so it is called 3DCNNLSTM (3-dimensional convolutional neural network with long short-term memory). Fig. 2 shows the sequence of layers that make up the architecture of 3DCNNLSTM.

At the entrance of 3DCNNLSTM there is a «conv1_1» convolutional layer. It accepts a sequence of 24 video frames of size 128×128, which are simultaneously processed by three-dimensional layer filters and pass through the ReLU activation function. As a result, the feature maps are formed, which arrive at the input of the next «conv1_2» convolutional layer, which forms new feature maps. In total, the network architecture contains 11 three-dimensional convolutional layers.

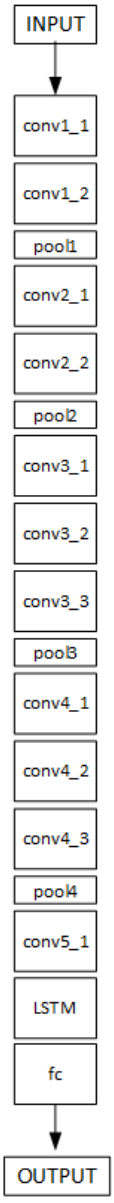


Fig. 2. The sequence of layers that make up the 3DCNNLSTM architecture

To reduce the spatial size of feature maps, as well as reduce the complexity of computations in the formation of feature maps and the probability of network retraining, sub-sampling layers are located between some of the convolutional layers. Arriving to these layers, feature maps are usually reduced in size due to the passage of a sliding window, which, using a special function, combines the captured values into one.

The 3DCNNLSTM architecture contains only 4 such layers, all of them use sliding windows of the same size, which move in the same step, and use the function of calculating the maximum value to combine the values. So, for example, 128 feature maps 128×128 in size and 24 in depth, falling from the «conv1_2» convolutional layer onto the sub-sampling layer «pool1, decrease and form 128 feature maps in the size 64×64 and depth 12.

The parameters of the described layers of 3DCNNLSTM model are presented in Table I.

TABLE I. CONVOLUTIONAL AND DOWNSAMPLING PARAMETERS OF 3DCNNLSTM MODEL

Layer number	Layer	Filter size [depth, height, width, in_channels, out_channels]	Sliding window dimensions [depth, height, width, channels]	Stride of sliding window [str_depth, str_width, str_height, str_channels]	Function	Size of feature maps to layer output [depth, height, width, channels]
1	conv1_1	[1, 2, 2, 3, 64]	-	[1, 1, 1, 1]	ReLU	[24, 128, 128, 64]
2	conv1_2	[1, 2, 2, 64, 128]	-	[1, 1, 1, 1]	ReLU	[24, 128, 128, 128]
3	pool1	-	[1, 2, 2, 2, 1]	[2, 2, 2, 1]	Max	[12, 64, 64, 128]
4	conv2_1	[1, 2, 2, 128, 256]	-	[1, 1, 1, 1]	ReLU	[12, 64, 64, 256]
5	conv2_2	[1, 2, 2, 256, 256]	-	[1, 1, 1, 1]	ReLU	[12, 64, 64, 256]
6	pool2	-	[1, 2, 2, 2, 1]	[2, 2, 2, 1]	Max	[6, 32, 32, 256]
7	conv3_1	[1, 2, 2, 256, 512]	-	[1, 1, 1, 1]	ReLU	[6, 32, 32, 512]
8	conv3_2	[1, 2, 2, 512, 512]	-	[1, 1, 1, 1]	ReLU	[6, 32, 32, 512]
9	conv3_3	[1, 2, 2, 512, 512]	-	[1, 1, 1, 1]	ReLU	[6, 32, 32, 512]
10	pool3	-	[1, 2, 2, 2, 1]	[2, 2, 2, 1]	Max	[3, 16, 16, 512]
11	conv4_1	[1, 2, 2, 512, 512]	-	[1, 1, 1, 1]	ReLU	[3, 16, 16, 512]
12	conv4_2	[1, 2, 2, 512, 512]	-	[1, 1, 1, 1]	ReLU	[3, 16, 16, 512]
13	conv4_3	[1, 2, 2, 512, 512]	-	[1, 1, 1, 1]	ReLU	[3, 16, 16, 512]
14	pool4	-	[1, 2, 2, 2, 1]	[2, 2, 2, 1]	Max	[2, 8, 8, 512]
15	conv5_1	[1, 2, 2, 512, 512]	-	[2, 1, 1, 1]	ReLU	[1, 8, 8, 512]

The table uses the following notation: depth – the depth of the sliding window; height – the height of the filter; width – the width of the filter; in_channels – the number of attribute cards arriving at the input; out_channels – the number of feature maps received at the output; channels – the number of channels covered by one sliding window. str_depth – step size of the sliding window in depth; str_width – step size of the sliding window wide; str_width – step size of the sliding window in height; str_channels – the number of channels covered by one sliding window.

Thus, from the initial sequence of 24 images of size 128×128, a set of 512 feature maps of size 8×8 is formed.

Next to the convolutional layers is the LSTM-layer (LSTM cell), it contains 512 memory blocks, the task of which is to preserve the state of the layer. Conventionally, this layer can be represented as a hidden layer in a fully connected network, as shown in Fig. 3.

For sequences fed to the input, the LSTM layer forms dependencies. In our case, this layer will form dependencies for a sequence consisting of 256 vectors of 128 values. Therefore, before entering the input of this layer, the feature maps are combined into a single array of size 128×128.

The processing of the input sequence by the LSTM layer can be expanded in time as shown in Fig. 4. The LSTM layer works as follows: at time t_1 a vector of 128 signs arrives at the input of the LSTM layer («FEATURE MAP ROW 1»), it is processed and stores a state inside, which will affect calculations inside the layer at time t_2. At time t_2, the next feature vector («FEATURE MAP ROW 2») arrives at the input, which, together with the state preserved in the layer at time t_1, determines the state of the LSTM layer at t_2, the same actions are performed at time t_3, etc. Thus, all 256 feature vectors are processed (that is, there will be 256 moments), each of which will determine the state of the layer, as a result, it will form 512 values at the output, which will be transmitted to the input of a fully connected network.

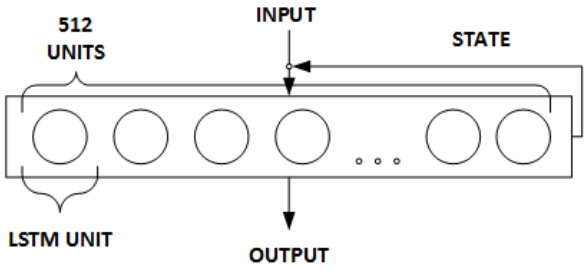


Fig. 3. LSTM cell

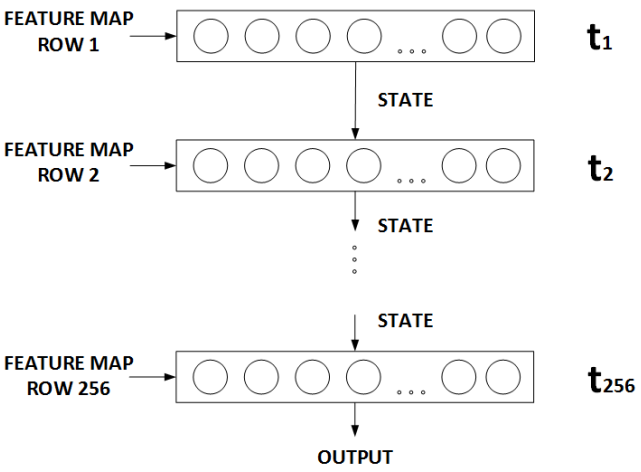


Fig. 4. Processing the input sequence of feature maps by LSTM cell

The LSTM model is being described by following an equations:

$$i_t = \sigma\left(x_t W_{xi} + h_{t-1} W_{hi} + c_{t-1} W_{ci} + b_i\right) \quad (1)$$

$$f_t = \sigma\left(x_t W_{xf} + h_{t-1} W_{hf} + c_{t-1} W_{cf} + b_f\right) \quad (2)$$

$$o_t = \sigma\left(x_t W_{xo} + h_{t-1} W_{ho} + c_{t-1} W_{co} + b_o\right) \quad (3)$$

$$c_t = f_t \circ c_{t-1} + i_t \circ \tanh\left(x_t W_{xc} + h_{t-1} W_{hc} + b_c\right) \quad (4)$$

$$h_t = o_t \circ \tanh\left(c_t\right) \quad (5)$$

where x_t is the input to the LSTM block, i_t, f_t, o_t, c_t, h_t are the input gate, the forget gate, the output gate, the cell state and the output of the LSTM block respectively at the current

time step t. L are the weights between the input layer and the input gate, the forget gate and the output gate respectively. W_{hf}, W_{hi}, W_{ho} are the weights between the hidden recurrent layer and the forget gate, the input gate and the output gate of the memory block respectively. W_{ci}, W_{cf}, W_{co} are the weights between the cell state and the input gate, the forget gate and the output gate respectively and finally, b_i, b_f, b_o are the additive biases of the input gate, the forget gate and the output gate respectively. The set of activation functions consists of the sigmoid function $\sigma()$, the element-wise multiplication $\circ()$ and the hyperbolic activation function $\tanh()$.

Four layers of a fully connected network are located behind the LSTM scrapping, shown in Fig. 5 («fc» block in Fig. 2). The first one has 512 neurons, it takes values from the LSTM layer. The next layer has 256 neurons and uses the ReLU activation function to calculate their values. Next comes a layer of 128 neurons, which uses the same activation function. The output network layer has 27 neurons (the number of video recording classes) and uses the softmax activation function to calculate their values.

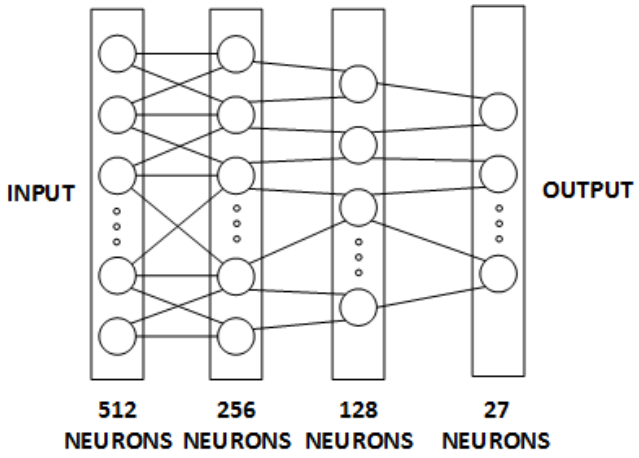


Fig. 5. Fully connected layers of 3DCNNLSTM model

IV. Data set. Baseline conditions

For training and validation of the neural network, the «The 20BN-jester Dataset V1» [23] dataset is selected. This data set includes 148,092 records representing video JPG frames, in which people in front of the camera of a laptop or a web camera make hands with predetermined gestures. In total, there are 25 classes with gestures: «Swiping Left», «Swiping Right», «Swiping Down», «Swiping Up», «Pushing Hand Away», «Stop Sign», etc., one class talking about the absence of a gesture («No gesture») and the class «Doing other things which says that some action is performed, but it cannot be assigned to any of the classes. The data set used is pre-classified and divided into data sets for training, testing and verification.

During preparation for training, records are formed into which 24 images of the initial set are recorded, whose dimensions are reduced to a scale of 128×128, each of which is assigned to one of 27 classes.

Before learning, the values of the filters of the convolutional layers of the neural network, LSTM blocks and the weights of the connections between the perceptron layers are initialized to default values, and at the end of each training epoch, the values obtained are written to the file system and the next epoch uses these values and learning continues.

V. Experiment. Training. Learning procedure

During the training, the records from the training set synchronously arrive at the input of the neural network. The entire learning process is logged and on each 1000th record, the accuracy of the trained model is checked on the data from the corresponding dataset.

Among the main metrics of the neural network there are: accuracy, precision, cross-entropy.

In our work, the cross entropy (values of which are also logged) serves to assess the accuracy of the model prediction. The closer the entropy value is to zero, the higher the prediction accuracy of the neural network. To minimize error of 3DCNNLSTM model, we use the Adam optimization algorithm [24], whose initial learning rate is 0.1, but with each epoch we reduce this value by an order of magnitude to more precisely adjust the model weights and reduce the cross-entropy value.

When evaluating a neural network, we also check its accuracy on a validation dataset. Accuracy determines whether the expected video recording class has the highest probability in predicting the trained model.

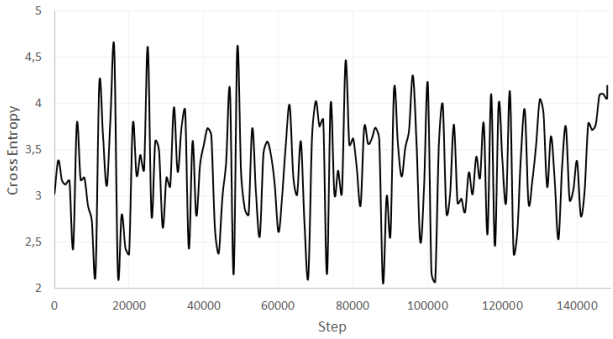
In order to avoid retraining inside the LSTM layer, a small error is added to the result produced by its blocks during training [25]. During validation the added error is zero and does not affect network prediction.

VI. Results. Productivity (time consuming, devices, resource costs)

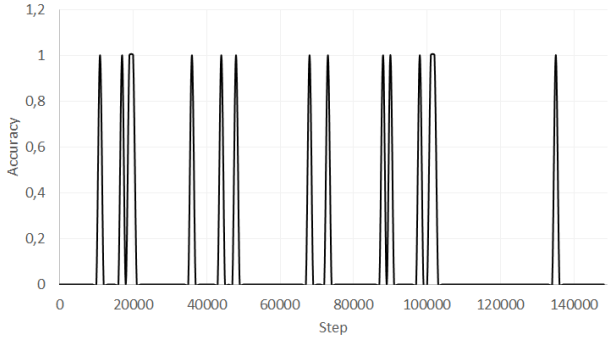
The 3DCNNLSTM model was trained on an NVIDIA Quadro M4000 graphics processor, with an average processing time of 1000 videos per 10 minutes, from which the average speed is 0.028 videos/second, or 0.67 frames/second.

During the first epoch of learning, when the learning rate is 0.1, the graph of the change in cross entropy and the prediction confidence on the validation data had the form shown in Fig. 6. As can be seen from the graphs, the cross entropy has a large amplitude and its average value is ~ 3.30, and the accuracy of the prediction only 15 times out of 144 tests gives a correct prediction.

During the second epoch of learning, when the learning rate is 0.01, the graph of the change in cross-entropy and the prediction accuracy on the validation data set was shown in Fig. 7. As can be seen from the first graph, the cross-entropy already has a smaller amplitude, and the accuracy of the prediction gives the correct prediction of 144 tests 16 times.

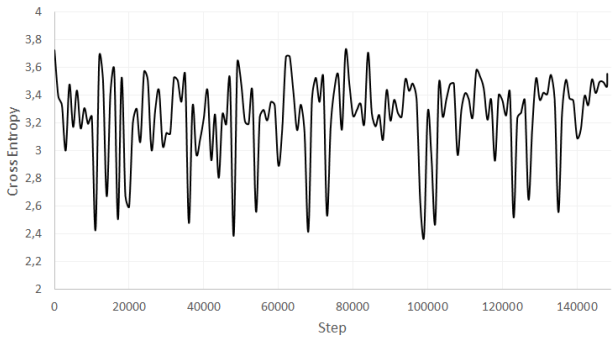


a)

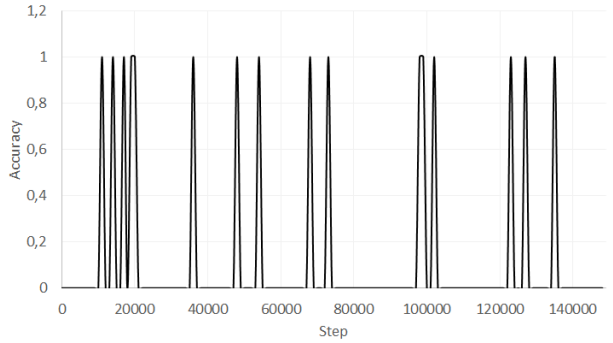


b)

Fig. 6. Values: a) cross entropy; b) accuracy of when 3DCNNLSTM model are validated with a learning rate of 0.1



a)

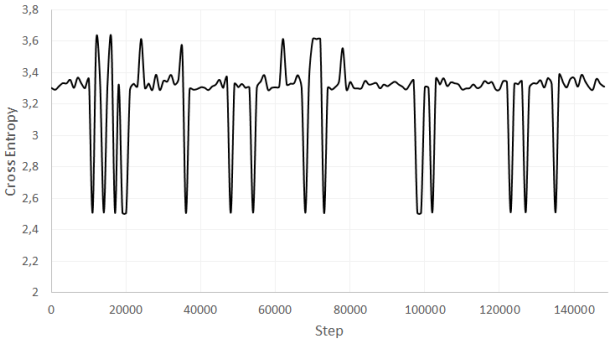


b)

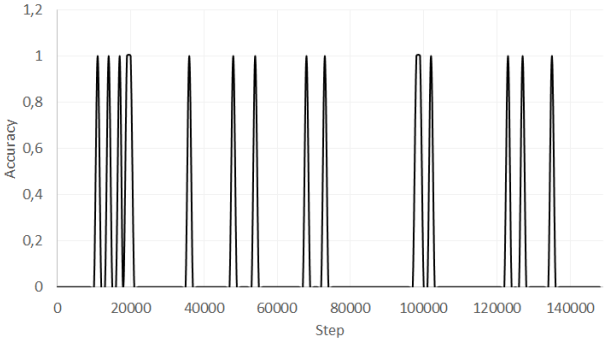
Fig. 7. Values: a) cross entropy; b) accuracy of prediction when 3DCNNLSTM model are validated with a learning rate of 0.01

Fig. 8 shows the result of the fifth learning era. At this era, the learning rate was 0.00001. From the graphs it is clear that the trained model assumes high accuracy in the same 16 examples of the test data set that were presented earlier, i.e. the accuracy of the model prediction is 11.11%, and the average value of cross entropy is ~3.30, which does not outperform state-of-the-art methods [17].

The graph of the cross-entropy change shows that the standard Adam optimization algorithm changes the values of weights, where the entropy periodically takes values that are lower, then higher than ~3.30, but does not lead the 3DCNNLSTM model to the required confidence indicators, according to which the average value of entropy should decrease with each epoch. In this regard, in the next versions of the model there will be an additional correction of the error before updating the weight values according to the algorithm that is currently at the development stage.



a)



b)

Fig. 8. Values: a) cross entropy; b) accuracy of prediction when 3DCNNLSTM model are validated with a learning rate of 0.00001

Thus, the developed model has overfitted and does not provide the expected prediction accuracy. To eliminate this problem, a decision was made to suspend network training and implement fine-tuning methods [26], which had not previously been adequately applied. Among them will be applied: initialization of the initial values of the weights using the Xavier or He method, batch normalization, data augmentation and ensembles. In addition, early stopping may be used, which relates to optimization of hyperparameters.

VII. Conclusion

The article describes the architecture and method of teaching the built 3DCNNLSTM model for recognizing simple dynamic gestures. When training a neural network, a set of data was used that contained 25 classes with simple gestures, but the gestures were recorded in color and with a complex, heterogeneous, and sometimes changing background. For these reasons, the results of training and validation gave a cross-sectional entropy average of around 3.30, which did not provide acceptable predictive accuracy (11.11%) to compete with modern neural networks (98.50% accuracy for the MSR-Gesture3D dataset and 98.20% accuracy for ChaLearn (Track 3) dataset). Nevertheless, this network is a promising basis for further developments related to the recognition of dynamic sequences of frames in a video stream. The network can be simplified, which will provide an increase in the speed of its work, and the presence of a memory unit in LSTM allows an increase the number of recognized classes. The 3DCNNLSTM model allows work not directly with images, but with the result of their processing, for example, Mask RCNN, and its input can be expanded to receive additional information from the depth sensor, for example, in addition to the already used RGB channels. In the new versions of the neural network, a special algorithm for minimizing cross entropy will be developed and fine-tuning methods will be applied.

References

[1] V. I. Voronov, K. V. Genchel, M. D. Artemov and D. N. Bezumnov, «"Surdotelephone" project with convolutional neural network», *2018 Systems of Signals Generating and Processing in the Field of on Board Communications*, Moscow, 2018, pp. 1-6.

[2] L. I. Voronova, R. V. Tolmachev and A. V. A. Usachev, «Resource development to prevent riots at mass events», *2018 Systems of Signals Generating and Processing in the Field of on Board Communications*, Moscow, 2018, pp. 1-5

[3] Mikhail D. Artemov, Lilia I. Voronova, «Design of a architecture of the program complex for equipment control using gestures», *Technology & Entrepreneurship in Digital Society (TEDS)*, Moscow, 2018

[4] A. Goncharenko, L. I. Voronova, V. I. Voronov, A. A. Ezhov and D. V. Goryachev, «Automated support system designing for people with limited communication», *2018 Systems of Signals Generating and Processing in the Field of on Board Communications*, Moscow, 2018, pp. 1-7

[5] A.S. Trunov, L. I. Voronova, V. I. Voronov, D. I. Sukhachev and V. G. Strelnikov, «Legacy applications model integration to support scientific experiment», *2018 Systems of Signals Generating and Processing in the Field of on Board Communications*, Moscow, 2018, pp. 1-7.

[6] Voronov Vyacheslav I., Genchel Ksenia V., Voronova Lilia I., Travina Maria D., «Development of a Software Package Designed to Support Distance Education fo0072 Disabled People», *IEEE-International Conference "2018 Quality Management, Transport and Information Security, Information Technologies" (IT&QM&IS-2018)*, St.Petersburg, pp.746-751 (in press)

[7] Mikhail D. Artemov, Vjacheslav.I.Voronov, Lilia I. Voronova, Artem A. Goncharenko, Vasiliy A.Usachev «Subsystem for Simple Dynamic Gesture Recognition Using 3DCNNLSTM», *24th Conference of Open Innovations Association FRUCT*, Moscow, Russia, 8-12 April 2019 (unpublished)

[8] Artem A. Goncharenko, Lilia I. Voronova , Mikhail D. Artemov, Vjacheslav.I.Voronov, Danil N.Bezumnov, «Sign Language Recognition Information System Development Using Wireless Technologies for People with Hearing Impairments», *24th Conference of Open Innovations Association FRUCT*, Moscow, Russia, 8-12 April 2019 (unpublished)

[9] Vyacheslav I. Voronov, Vladimir G. Strelnikov, Liliya I. Voronova, Artyom S. Trunov, Andrey Vovik "Faces 2D-Recognition and Identification Using the HOG Discriptors Method" 24th Conference of Open Innovations Association FRUCT, Moscow, Russia, 8-12 April 2019 (unpublished)

[10] «Gesture recognition using end-to-end learning from a large video database. 2017», medium.com, 2017, https://medium.com/twentybn/gesture-recognition-using-end-to-end-learning-from-a-large-video-database-2ecbfb4659ff

[11] Eleni Tsironi, Pablo Barros и Stefan Wermter, «Gesture Recognition with a Convolutional Long Short-Term Memory Recurrent Neural Network», *Proceedings of the European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN)*, Hamburg – Germany, 2016, pp. 213-218, https://www2.informatik.uni-hamburg.de/wtm/ps/Tsironi_ESANN_2016.pdf

[12] Christopher Olah, «Understanding LSTM Networks», 2015, http://colah.github.io/posts/2015-08-Understanding-LSTMs/

[13] Núñez Fernández, Dennis & Kwolek, Bogdan, «Hand Posture Recognition Using Convolutional Neural Network», 2017, http://home.agh.edu.pl/~bkw/research/pdf/2017/FernandezKwolek_CIARP2017.pdf

[14] Limonchik, Ben, «3 D Model-Based Data Augmentation for Hand Gesture Recognition», 2017, http://cs231n.stanford.edu/reports/2017/pdfs/218.pdf

[15] Pinzón Arenas, Javier & Moreno, Robinson & Useche Murillo, Paula, «Hand gesture recognition by means of region-based convolutional neural networks», *Contemporary Engineering Sciences*, 2017, 10. 1329-1342. 10.12988/ces.2017.710154.

[16] G. Devineau, F. Moutarde, W. Xi and J. Yang, «Deep Learning for Hand Gesture Recognition on Skeletal Data», *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*, Xi'an, 2018, pp. 106-113

[17] Maryam Asadi-Aghbolaghi, Albert Clapes, Marco Bellantonio, Hugo Jair Escalante, Víctor Ponce-López, Xavier Baró, Isabelle Guyon, Shohreh Kasaei, Sergio Escalera, «A Survey on Deep Learning Based Approaches for Action and Gesture Recognition in Image Sequences», *12th IEEE International Conference* on, 2017, pp.476-483, https://hal.inria.fr/hal-01668383

[18] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, Manohar Paluri, «Learning Spatiotemporal Features with 3D Convolutional Networks», 2015, https://arxiv.org/abs/1412.0767

[19] Pavlo Molchanov, Shalini Gupta, Kihwan Kim, and Jan Kautz, «Hand Gesture Recognition with 3D Convolutional Neural Networks», *2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Boston, 2015, pp. 1-7, https://www.cv-foundation.org/openaccess/content_cvpr_workshops_2015/W15/papers/Molchanov_Hand_Gesture_Recognition_2015_CVPR_paper.pdf

[20] Otkrist Gupta, Dan Raviv, Ramesh Raskar, «Multi-velocity neural networks for gesture recognition in videos», 2016, https://arxiv.org/pdf/1603.06829.pdf

[21] Pichao Wang, Wanqing Li , Song Liu, Zhimin Gao, Chang Tang and Philip Ogunbona, «Large-scale Isolated Gesture Recognition Using Convolutional Neural Networks», 2017: https://arxiv.org/pdf/1701.01814.pdf

[22] Yang, Xiaodong & Molchanov, Pavlo & Kautz, Jan, «Making Convolutional Networks Recurrent for Visual Sequence Learning», 2018, 6469-6478. 10.1109/CVPR.2018.00677

[23] Dataset «The 20BN-jester Dataset V1», https://20bn.com/datasets/jester

[24] Diederik P. Kingma, Jimmy Ba, «Adam: A Method for Stochastic Optimization», *3rd International Conference for Learning Representations*, San Diego, 2015, pp. 1-15, https://arxiv.org/abs/1412.6980

[25] Yarin Gal, Zoubin Ghahramani, «A Theoretically Grounded Application of Dropout in Recurrent Neural Networks», NIPS 2016, 2016, https://arxiv.org/abs/1512.05287

[26] Blog of company «Wunder Fund», habr.com, «Глубокое обучение для новичков: тонкая настройка нейронной сети», 2016, https://m.habr.com

[27] ru/company/wunderfund/blog/315476/