Three-Dimensional Convolutions and Temporal Data for Sign Language Recognition

Serhii Kondratiuk[a,b], Iurii Krak[a,b], Vladislav Kuznetsov[a] and Anatoliy Kulias[a]

[a] *Glushkov Cybernetics Institute, Kyiv, 40, Glushkov ave., 03187, Ukraine*
[b] *Taras Shevchenko National University of Kyiv, Kyiv, 64/13, Volodymyrska str., 01601, Ukraine*

Abstract

The technology is proposed for recognition of gesture units (fingerspelling alphabet) of sign language. Implemented technology performs recognition of dactyl items from camera input using trained on collected training dataset set convolutional neural network, based on the MobileNetv2 architecture with spatio-temporal overlapping approach. Multiple configurations were used and based on experiments, optimal configuration in terms of complexity and quality was selected. On the test dataset accuracy of over 96% is achieved.

Keywords [1]

Sing language, recognition, convolutional neural network, mobilenetv2.

1. Introduction

Sign language is a widely spread mean of communicating among people with special communication requirements. In order to connect with society and within their own group, those who have hearing impairments might make use of supplementary software. The dactyl alphabet should be learned using gesture recognition technology, which would be included in this information technology. Smartphones, along with personal computers and laptops, have risen in popularity as devices with operating systems in recent years. Because it enables the technology to be developed and operated without modifying the code, cross-platforming is critical because it gives users a consistent experience across a variety of platforms, including mobile, low-resource, and powerful, as well as stationary. Gestural recognition is increasingly used in fields like as communication, human-computer interactions, etc. When it comes to platform variety, one solution is to use distributed computing and cross-platform programming [1, 2]. Instead of using virtual machines [3] or doing a lot of mono-platform programming, consider using cross-platform development [4, 5]. Using machine learning methods and neural networks, the study attempts to recognize sign language gestures and construct cross-platform modules that can operate on a range of current devices. Single gesture communication technology includes sing (gesture) recognition, and this article builds on past work by the author [6, 7, 8].

2. Existing approaches

Hand gesture detection may be seen as a form of object recognition challenge that has a number of mature and unique algorithms in conventional computer vision as well as deep learning, including convolution neural networks in particular.

Convolutional neural networks with 3-dimensional convolutions became effective when larger datasets with recorded activities were available (AlexNet [9], Sports-1M [10], Kinetics [11], Jester [12]). Because of the dataset's size, we were able to train the model without worrying about it being overfit [13]. Various algorithms based on conventional computer vision with hand-crafted features,

Information Technology and Implementation (IT&I-2021), December 01-03, 2021, Kyiv, Ukraine
EMAIL: sergey.kondrat1990@gmail.com (S.Kondratiuk); krak@univ.kiev.ua (I.Krak); kuznetsowwlad@gmail.com (V.Kuznetsov); anatoly016@gmail.com (A.Kulias)
ORCID: 0000-0002-5048-2576(S.Kondratiuk); 0000-0002-8043-0785(I. Krak); 0000-0002-1068-769X(V. Kuznetsov); 0000-0003-3715-1454 (A.Kulias)

such as the orientation of histograms [4], the histogram of oriented gradients (HOG) [14], bag-of-features [15], hyperplanes separation [16] were used to recognize sign language gestures. As with other computer vision applications, current state-of-the-art hand gesture recognition architectures [17, 18, 19] are based on CNNs.A focus on lightweight architectures, such as SqueezeNet [22], MobileNet [23], MobileNetV2 [24], ShuffleNet [25] and ShuffleNetV2 [26], MobileNetV3 [27], which seek to minimize computational cost while maintaining high accuracy, was made in research on current methodologies among CNN [20, 21]. MobileNetV2's 2D and 3D versions have both been utilized in our projects.

3. Problem statement

The suggested system should include a module for recognizing sign language gestures. Modules should be cross-platform compatible and execute without the need for codebase change on different systems. The gesture recognition module should have a model that can detect and recognize the gesture provided by the user from a camera input. Gestures are constrained by the Ukrainian dactyl language, although they may be expanded. To evaluate the model's performance, gather a suitable dataset of Ukrainian dactyl language [28]. For high accuracy and FPS-rate on many platforms, employing cross-platform technologies, the gesture recognition module should use a model that shows resilient and state-of-the-art performance as well as high efficiency in terms of processing resources.

4. Proposed approach

Proposed approach suggests using cross-platform technologies to create Ukrainian dactyl language recognition software that can work on several operating systems without modifying the code base. Tensorflow [29, 30] is suggested as a cross-platform framework for developing a gesture recognition module. By using a cross-platform machine learning framework, a gesture recognition model may be constructed and trained just once, and then deployed across different platforms (mobile, desktop, and online) with no need to change the model or training code. It's a unified cross-platform technology for Ukrainian dactyl language recognition with upgraded MobileNet architecture for better recognition of the Ukrainian dactyl alphabet all in all, which is the innovation suggested for the technology.

5. Gesture recognition

Cross-platform tools should be used to create gesture recognition for Ukrainian dactyl language recognition as part of cross-platform technology. Ong et al. [31] offer Sequential Pattern Mining for the detection of indications based on the tree topologies in their technique.

In the field of image and video analysis, convolutional neural networks (CNNs) are most typically used as regularized versions of multilayer perceptron. CNNs excel in image analysis due to their ability to take into consideration the picture's location reference for the data they process (typically nearby samples at some input data are not related, which is not true in case of an image). As a consequence, CNN's picture classification and recognition findings are cutting-edge.

In addition to recognizing characteristics, dynamic gesture recognition requires modeling the time component of motions. Spatio-temporal classifiers can recognize sequences of spatial descriptors or pictures by creating descriptors that incorporate both spatial and temporal information.

It was decided to go with this technique in this case study. If the video stream has to be analyzed, either a single gesture picture or a succession of images may be used as input data for the neural network. It's possible to train the network to be more resistant to change in a dynamic object like the hand by looking at numerous surrounding photos concurrently. This enables the network to be trained to take into account the temporal element, i.e. the dynamics of change in motions in several photographs. If a picture has artifacts, has poor lighting, is blurry, or is occluded in some way, gesture recognition may be used to smooth things out using neighboring frames in succession.

6. Spatio-temporal approach

The usage of a temporary floating window was suggested and executed to improve the efficiency of this strategy. To achieve this, the input sequence should be divided into n subsequences, each having a

minimum length of m, and these subsequences should all overlap at some point (into a certain part, from 10 percent to 50 percent of the subsequence length) as shown at Fig. 1.

It is possible for the same fingerprints to have different exterior characteristics on the hand (the challenge of identifying these differences falls on the recognition model) and different data parameters while acquiring video sequences and individual frames, as well (size, quality, focal length, lighting, background, artifacts, blur and etc.). A uniform data processing approach was designed to convert them to a generic form for further computations inside the specified recognition model, both at the training and recognition stages.

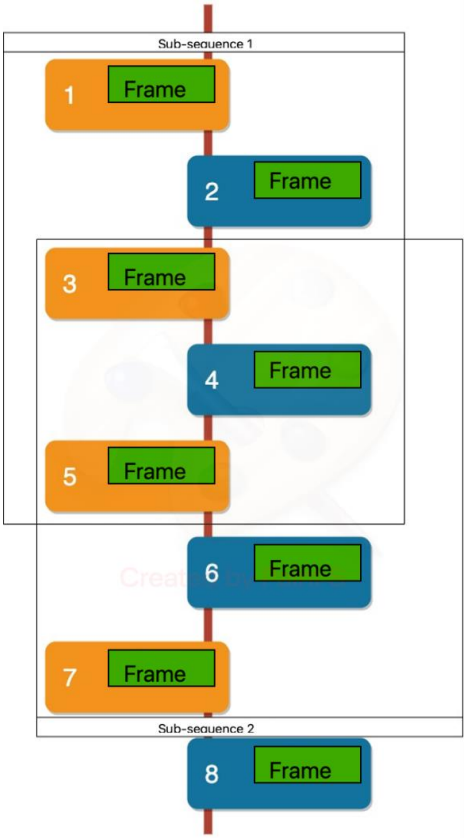

Figure 1: Sequence of frames, divided into two sub-constants of five frames, which intersect into three frames

Thus, from one input video stream with a gesture you can get *n* video streams of this gesture of a smaller size. Therefore, the data can be presented as:

$$D = \{d_{i-k},...,d_i,...,d_{i+k}\},\ \ i = \overline{1,n-k}, \tag{1}$$

where *k* is the number of previous and subsequent frames from the current, from which a sequence of images is formed (Fig. 2).

Adapting the neural network to the input data's space-time format is the goal of this article. Three-dimensional convolutions were introduced to enhance the architecture of the convolutional neural network in order to better use the input data's spatiotemporal properties. Convolution may be performed in picture space as well as time, which makes it possible to use three-dimensional convolutional neural networks for this purpose.

There are three stages to data processing:

- normalizing
- noise reduction
- shrinkage

to a single size are all examples of normalization.

The new MobileNetV2 architecture (Fig. 3) is a development of the MobileNet concept and is a new mobile architecture. MobileNetV2 adds two new features over its predecessor. Residual blocks use a skip connection to link the beginning and finish of a convolution block. The addition of these two states

allows the network to retrieve activations from previous blocks that weren't altered throughout the convolutional process.

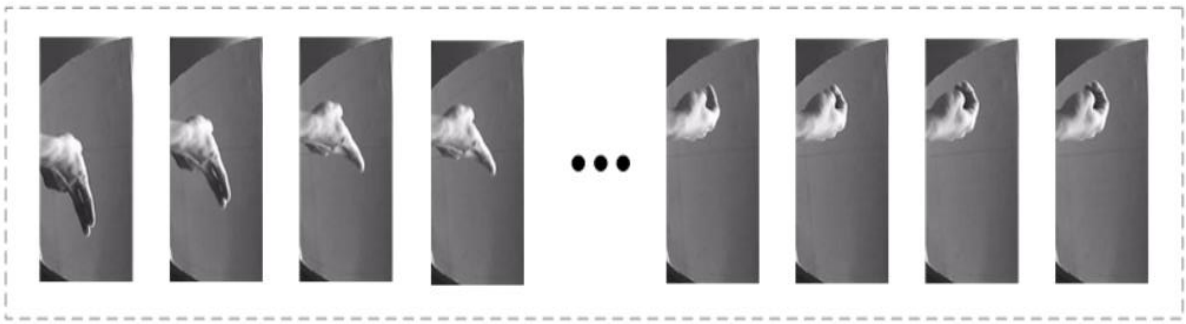

Figure 2: Two subsequences created from a single video stream

Input	Operator	exp size	$\#out$	SE	NL	s
$224^2 \times 3$	conv2d, 3x3	-	16	-	HS	2
$112^2 \times 16$	bneck, 3x3	16	16	✓	RE	2
$56^2 \times 16$	bneck, 3x3	72	24	-	RE	2
$28^2 \times 24$	bneck, 3x3	88	24	-	RE	1
$28^2 \times 24$	bneck, 5x5	96	40	✓	HS	2
$14^2 \times 40$	bneck, 5x5	240	40	✓	HS	1
$14^2 \times 40$	bneck, 5x5	240	40	✓	HS	1
$14^2 \times 40$	bneck, 5x5	120	48	✓	HS	1
$14^2 \times 48$	bneck, 5x5	144	48	✓	HS	1
$14^2 \times 48$	bneck, 5x5	288	96	✓	HS	2
$7^2 \times 96$	bneck, 5x5	576	96	✓	HS	1
$7^2 \times 96$	bneck, 5x5	576	96	✓	HS	1
$7^2 \times 96$	conv2d, 1x1	-	576	✓	HS	1
$7^2 \times 576$	pool, 7x7	-	-	-	-	1
$1^2 \times 576$	conv2d 1x1, NBN	-	1024	-	HS	1
$1^2 \times 1024$	conv2d 1x1, NBN	-	k	-	-	1

Figure 3: Architecture of MobileNetv2

There have been two strategies to improve the network:

- swish non-linearity
- layer removal

Using the developed approach of accumulation of probability from prior subsequences, the research suggests a methodology for smoothing anomalous recognition results. Using a floating window with intersections, subsequences emerge on the premise of reducing the maximum probability of a gesture and increasing the maximum probability of the following gesture over time. Predictions from earlier subsequences are used to build up the model, which then uses that information to update the current recognition result only when the total number of predictions surpasses a certain threshold.

$$\sum_{t=t-n}^{t+n} \sum_{i=t-k}^{t+k} p_i > threshold\ , \tag{2}$$

where: p_i - the probability of a gesture in the frame; i - frame number on the current subsequence; t - number of the current subsequence; k - the size of the subsequence in both directions; n - number of accumulated subsequences. By using a projection layer on top of the preceding block's final layer, this was accomplished. As a result, the preceding bottleneck layer's projection and filtering layers may be removed (block).

7. Ukrainian dactyl dataset collection for gesture recognition using MobileNet

An examination of the educational data collection gathered for the first time by people and the environment for the Ukrainian dactyl alphabet in such amount and variety (Fig. 4). We experimented

with various lighting settings (with distribution: 20 percent of images in low light conditions, 30 percent in low light conditions and 50 percent in high quality lighting). Noisy and blurry pictures accounted for around 10 percent of the total pictures in the collection. A training data set of around 50,000 original photos was produced.

Figure 4: Dataset example

Additional data augmentation methods (such as rotation, random cropping, mirroring, and so on) resulted in a final data collection of around 150,000 pictures. After selecting a tenth of the whole data set for testing, we had 135,000 photos and 15,000 images for final testing.

Augmenting the amount of an existing dataset without having to manually create additional photos is known as data mining. There are various strategies for augmenting data, and they all include increasing the quantity of pictures and diversifying them while also making it less likely for the neural network to overfit characteristics seen in the original data set.

The original data set may be further distorted by combining image alteration techniques. As a result, the conditions under which the trained model is evaluated may be altered.

As a result of the data growth, procedures like the following were performed (Fig. 5): Gaussian noise; affine transformation; trimming + shift; reflection; distortion of perspective; blurring.

During dataset collection it is important to maintain it's statistically significant diversity in the data, but maintain similar distribution among train and test data – not to introduce unwanted bias for the model, which is present in train data but absent in test data. This would lead the model into worse performance in real life cases, whilst showing artificially better performance during testing.

For instance, Fig. 6 shows distribution in light condition in the train and test split of the dataset, represented in three types: poor light condition, mediocre light condition and good light condition.

8. Experiments

The software implementation of dactyl recognition of the Ukrainian dactyl alphabet was put to the test using a variety of different techniques. Several adjustments to the design were made throughout the Convolutional Neural Network training process based on the MobileNetv2 architecture, which shows good quality and performance on mobile and devices with low computational capacity.

The training architecture may be adjusted using hyperparameters (learning rate, batch size, number of epochs) and the architecture itself (number and configuration of repeating the same kind of layers), which are picked for each training independently.

Five distinct neural network architecture configurations were constructed using the developed technology, each one with varying numbers of layers and parameters, allowing for a balanced neural network design that was both small and effective on the test data set.

The trained model's accuracy plateaued with time, as seen in Fig. 7, therefore the architecture No. 4 (Table 1) was selected as the best option since it was the smallest and had the highest accuracy (average macro-score f1). It is important to analyze confusion matrix of model prediction (Fig. 8). This also helped to select the best approach and best configuration. In order to train and compare neural networks on a single test set, a grid of alternative configurations and hyperparameter values is created.

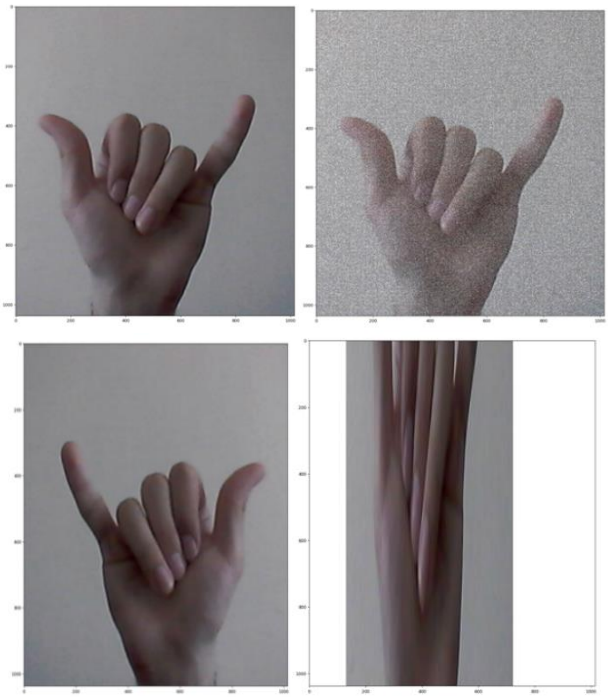

Figure 5: Original image (topmost) and augmented images.

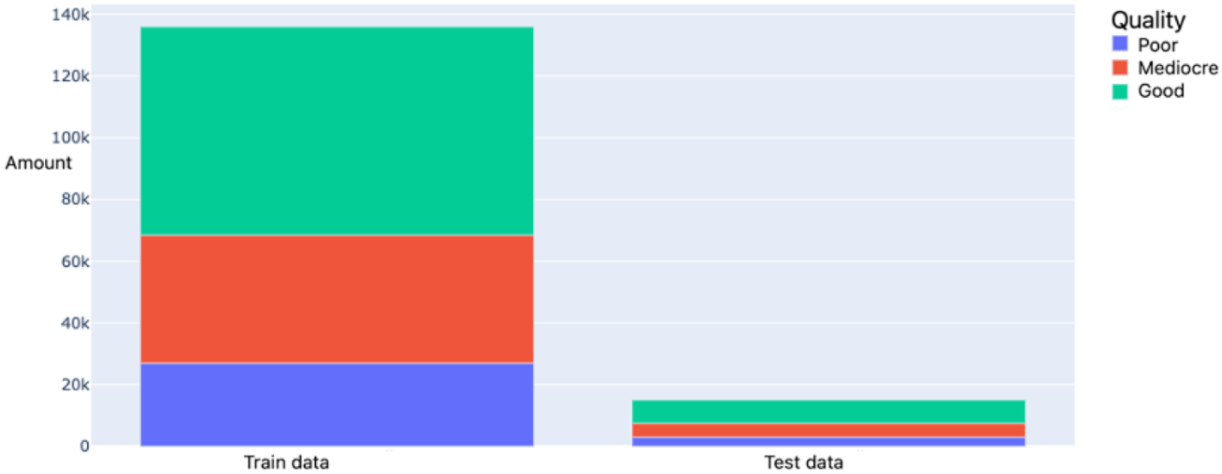

Figure 6: Distribution of light quality on the train and test datasets.

The chosen MobileNetv2 architecture, which demonstrates high quality and performance on mobile devices and devices with limited computing power, can, however, at the training stage be configured with hyperparameters that are selected for each training individually (learning rate, batch size, number of epochs), and the architecture itself, i.e. the number and configuration of repeating layers of the same type. This set of configurations and possible values of hyperparameters forms a grid within which a set of neural networks is trained and compared on a single test set.

Each training was subjected to standard strategies for combating neural network overfitting. Model's prediction time is sufficient for real-time (24 fps) performance using Nvidia K80 GPU.

Model performance based on architecture

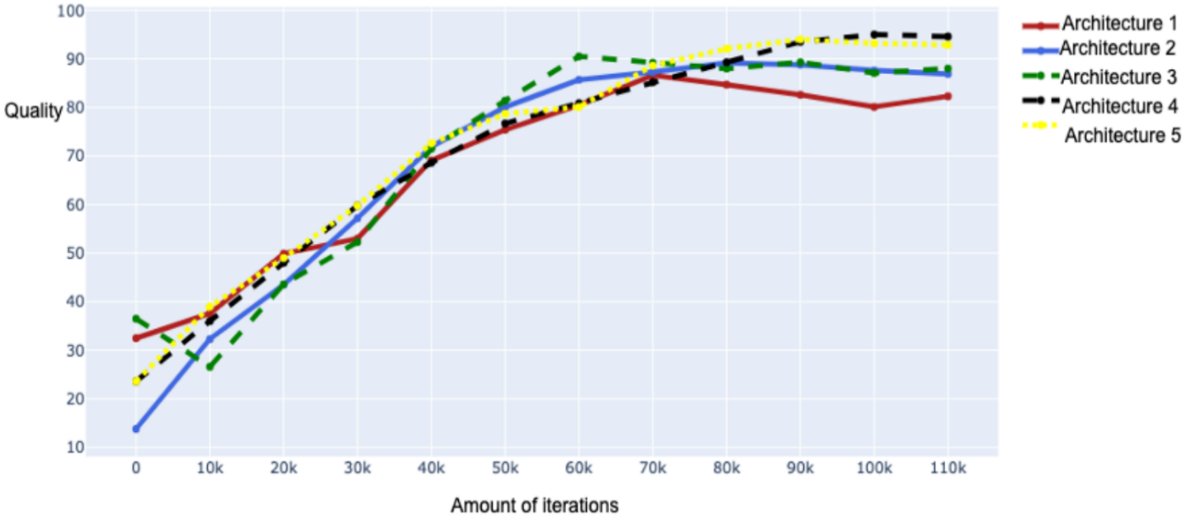

Figure 7: Model quality related to architecture and number of iterations

Table 1: Architectures considered

Architecture 1	Architecture 2	Architecture 3	Architecture 4	Architecture 5
Conv / s2	Conv / s2	Conv / s2	Conv / s2	Conv / s2
Conv dw / s1	Conv dw / s1	Conv dw / s1	Conv dw / s1	Conv dw / s1
Conv / s1	Conv / s1	Conv / s1	Conv / s1	Conv / s1
Conv dw / s2	Conv dw / s2	Conv dw / s2	Conv dw / s2	Conv dw / s2
Conv / s1	Conv / s1	Conv / s1	Conv / s1	Conv / s1
Conv dw / s1	Conv dw / s1	Conv dw / s1	Conv dw / s1	Conv dw / s1
Conv / s1	Conv / s1	Conv / s1	Conv / s1	Conv / s1
Conv dw / s2	Conv dw / s2	Conv dw / s2	Conv dw / s2	Conv dw / s2
Conv / s1	Conv / s1	Conv / s1	Conv / s1	Conv / s1
Conv dw / s1	Conv dw / s1	Conv dw / s1	Conv dw / s1	Conv dw / s1
Conv / s1	Conv / s1	Conv / s1	Conv / s1	Conv / s1
Conv dw / s1	2 x Conv dw / s1	3 x Conv dw / s1	Conv dw / s2	Conv dw / s2
Conv / s1	2 x Conv / s1	3 x Conv / s1	Conv / s1	Conv / s1
Conv dw / s2	Conv dw / s2	Conv dw / s2	4 x Conv dw / s1	5 x Conv dw / s1
Conv / s1	Conv / s1	Conv / s1	4 x Conv / s1	5 x Conv / s1
Avg Pool / s1	Avg Pool / s1	Avg Pool / s1	Conv dw / s2	Conv dw / s2
FC / s1	FC / s1	FC / s1	Conv / s1	Conv / s1
Softmax / s1	Softmax / s1	Softmax / s1	Avg Pool / s1	Conv dw / s2
			FC / s1	Conv / s1
			Softmax / s1	Avg Pool / s1
				FC / s1
				Softmax / s1

Example grid:

$$\begin{cases} \text{learning\_rate: } [0.001, 0.0001], \\ \text{batch\_size: } [8,16,32], \\ \text{layers\_config: } [\text{config1, config2,config3}] \end{cases} \quad (3)$$

9. Conclusions

There is a core gesture recognition module in the relational database that uses the database containing gesture specifications provided in YAML format.

A unique model architecture and data preparation processes are required to enhance outcomes for gesture detection in a video, according to proven data processing techniques.

It was shown that adopting the sophisticated MobileNetv2 architecture with three-dimensional convolution and spatio-temporal overlapping subsequences improved the quality of recognition when compared to studies using other model architectures and data sets. As a result of this selection

procedure, we know that the model's complexity and recognition efficiency are optimally aligned. On a specific test set, the model's quality was reached at 0.96 macro-score f1.

A picture data collection containing all 50 Ukrainian dactyls shown by 50 distinct persons was gathered for the first time as part of the suggested implementation and up to 150,000 photos were enhanced. Other gestures and languages, as well as cross-platform modules, may be added to the proposed gesture communication system.

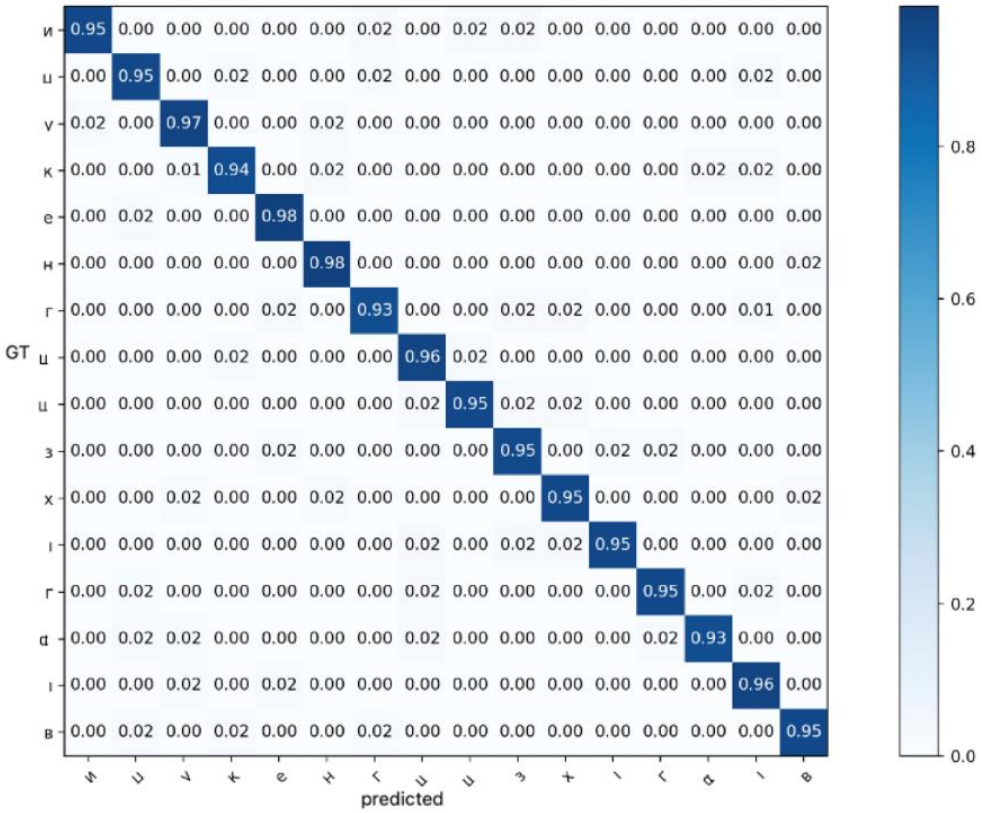

Figure 8: Confusion matrix of architecture # 4.

10. References

[1] P. Mell, T. Grance (September 2011). The NIST Definition of Cloud Computing (Technical report). National Institute of Standards and Technology: U.S. Department of Commerce. doi:10.6028/NIST.SP.800-145. Special publication 800-145

[2] The Linux Information Project, Cross-platform Definition. www.linfo.org

[3] Yu.V. Krak, A.V. Barmak, E.M. Baraban. Usage of nurbs-approximation for construction of spatial model of human face. Journal of Automation and Information Sciences. 43(2) (2011): 71-81. doi:10.1615/JAutomatInfScien.v43.i2.70

[4] W.T. Freemanand, M. Roth. Orientation histograms for hand gesture recognition. In International workshop on automatic face and gesture recognition. volume 12, pages 296–301, 1995.

[5] J. Smith, N. Ravi. The Architecture of Virtual Machines. Computer. IEEE Computer Society. 38 (5) (2005): 32–38.

[6] S. Kondratiuk, I. Krak. Dactyl Alphabet Modeling and Recognition Using Cross Platform Software, Proceedings of the 2018 IEEE 2nd International Conference on Data Stream Mining and Processing, DSMP 2018, pp. 420-423. doi: 10.1109/DSMP.2018.8478417

[7] Yu.V. Krak, Yu.V. Barchukova, B.A. Trotsenko. Human hand motion parametrization for dactylemes modeling, Journal of Automation and Information Sciences, 43(12) (2011):1-11. doi:10.1615/JAutomatInfScien.v43.i12.10

[8] I.G. Kryvonos, I.V. Krak. Modeling human hand movements, facial expressions, and articulation to synthesize and visualize gesture information, Cybernetics and Systems Analysis: 47(4) (2011): 501-505. doi: 10.1007/s10559-011-9332-4

[9] A. Krizhevsky, I. Sutskever, G.E. Hinton. Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems, pages 1097-1105, 2012.
[10] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, L. Fei-Fei. Large-scale video classification with convolutional neural networks. In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, pages 1725-1732, 2014.
[11] J. Carreira, A. Zisserman. Quovadis, action recognition a new model and the kinetics dataset. In Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference, pages 4724-4733. IEEE, 2017.
[12] T. B. N. GmbH. The 20bn-jester dataset v1. https://20bn.com/datasets/jester, 2019.
[13] K. Hara, H. Kataoka, Y. Satoh. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, pages 18-22, 2018.
[14] L. Prasuhn, Y. Oyamada, Y. Mochizuki, H. Ishikawa. A hog-based hand gesture recognition system on a mobile device. In 2014 IEEE International Conference on Image Processing (ICIP), pages 3973-3977. IEEE, 2014.
[15] N. H. Dardas, N. D. Georganas. Real-time hand gesture detection and recognition using bag-of-features and support vector machine techniques. IEEE Transactions on Instrumentation and measurement, 60(11) (2011): 3592-3607.
[16] I.V. Krak, G.I. Kudin, A.I. Kulias. Multidimensional Scaling by Means of Pseudoinverse Operations, Cybernetics and Systems Analysis, 55(1) (2019):22-29. doi: 10.1007/s10559-019-00108-9
[17] O. Kopuklu, N. Kose, G. Rigoll. Motion fused frames: Data level fusion strategy for hand gesture recognition. arXiv preprint arXiv:1804.07187, 2018.
[18] P. Molchanov, S. Gupta, K. Kim, K. Pulli. Multi-sensor system for driver's hand-gesture recognition. In Automatic Face and Gesture Recognition (FG), 2015 11th IEEE International Conference and Workshops, volume 1, pages 1-8. IEEE, 2015.
[19] P. Molchanov, S. Gupta, K. Kim, J. Kautz. Hand gesture recognition with 3d convolutional neural networks. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, pages 1-7. June 2015.
[20] J. Hu, L. Shen, G. Sun. Squeeze-and-excitation networks. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 7132-7141, 2018.
[21] K. He, X. Zhang, S. Ren, J. Sun. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 770-778, 2016.
[22] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, K. Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and 0.5 mb model size. arXiv preprint arXiv:1602.07360, 2016.
[23] A.G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861, 2017.
[24] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, L.-C. Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 4510-4520. IEEE, 2018.
[25] X. Zhang, X. Zhou, M. Lin, J. Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 6848–6856. IEEE, 2018.
[26] N. Ma, X. Zhang, H.-T. Zheng, J. Sun. Shufflenet v2: Practical guidelines for efficient cnn architecture design. arXiv preprint arXiv:1807.11164, 5, 2018.
[27] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang. Searching for MobileNetV3. axXiv: 1905.02244, 5, 2019
[28] ASL Sing language dictionary [http://www.signasl.org/sign/model]
[29] Unity3D framework [https://unity3d.com/]
[30] Tensorflow framework documentation [https://www.tensorflow.org/api/]
[31] Eng-Jon Ong et al. Sign language recognition using sequential pattern trees. In: Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on. IEEE. 2012, pp. 2200-2207