INTL JOURNAL OF ELECTRONICS AND TELECOMMUNICATIONS, 2019, VOL. 65, NO.2 , PP. 303-308
Manuscript received January 14, 2019; revised April, 2019. DOI: 10.24425/ijet.2019.126314

Recognition of Sign Language from High Resolution Images Using Adaptive Feature Extraction and Classification

Filip Csóka, Jaroslav Polec, Tibor Csóka, Juraj Kačur

***Abstract*—A variety of algorithms allows gesture recognition in video sequences. Alleviating the need for interpreters is of interest to hearing impaired people, since it allows a great degree of self-sufficiency in communicating their intent to the non-sign language speakers without the need for interpreters. State-of-the-art in currently used algorithms in this domain is capable of either real-time recognition of sign language in low resolution videos or non-real-time recognition in high-resolution videos. This paper proposes a novel approach to real-time recognition of fingerspelling alphabet letters of American Sign Language (ASL) in ultra-high-resolution (UHD) video sequences. The proposed approach is based on adaptive Laplacian of Gaussian (LoG) filtering with local extrema detection using Features from Accelerated Segment Test (FAST) algorithm classified by a Convolutional Neural Network (CNN). The recognition rate of our algorithm was verified on real-life data.**

***Keywords*—sign language, gesture, sign, recognition, CNN, LoG, real-time, pattern recognition, machine learning**

I. Introduction

EXISTING state-of-the-art image recognition algorithms struggle in real-life applications, especially when it comes to high resolution images produced by devices people nowadays carry with them daily. Most of the recent advancements rely almost exclusively either on deep neural networks [1] or algorithms with high computational requirements such as SIFT or SURF [2]. All of these methods suffer from real-time processing performance limitations, resulting in the need to greatly reduce the resolution (via under-sampling) of images used for recognition. This effectively negates all benefits the increased image resolution brings to the process of recognition and limits achievable recognition rate [3], [4]. To this day there is not a single publicly available proven system for ASL recognition from video sequences that would work on devices with computational power like smart mobile phones or standard desktop computers which would use only input images for real-time recognition. The need for such systems arises from social demand for communication means usable by people with hearing impairments. They are often pushed from the mainstream society due to not being able to effectively communicate with anyone but members of their community and their own family. Having the ability to be able to understand what a person using ASL wants to convey just by pointing a camera of a regular smartphone and its standard computational power is highly desirable.

This work was supported by VEGA 1/0440/19 research grant.

Authors are with Institute of Multimedia Information and Communication Technologies - Faculty of Electrical Engineering and Information Technology STU in Bratislava, Slovakia (e-mail: xcsokaf@stuba.sk, polec@ktl.elf.stuba.sk, csokat@gmail.com, juraj.kacur@stuba.sk).

II. Theory

Our method is based on finding features in images using adaptive LoG filtering, followed by local extrema detection using method FAST-like algorithm. Extracted features are used as input into Convolutional Neural Network.

A. Laplacian of Gaussian

The Laplacian of Gaussian is yielded by the second derivative of a standard Gaussian function [5]:

$$LoG(x,y) = \nabla^2 G_\sigma(x,y) = \frac{\partial^2 G_\sigma(x,y)}{\partial x^2} + \frac{\partial^2 G_\sigma(x,y)}{\partial y^2} \quad (1)$$

Where G_σ is a Gaussian defined as [5]:

$$G_\sigma(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \quad (2)$$

Applying the derivations yields the LoG as [5]:

$$LoG(x,y) = -\frac{1}{\pi\sigma^4}\left[1 - \frac{x^2+y^2}{2\sigma^2}\right] e^{-\frac{x^2+y^2}{2\sigma^2}} \quad (2)$$

This filter has the advantage of preforming smoothing and high frequency edge detection at the same time, thereby suppressing the effects of high frequency noise [6], [7].

B. Features from accelerated segment test

The original FAST algorithm was designed for identifying points of interest in images. The algorithm calculated a Bresenham circle of radius 3 for each point in an image [8] and each pixel in the circle is clockwise assigned an addressing integer number from the range 1-16 [14]. If a set of N contiguous pixels in the circle have greater intensity value than the intensity of a candidate pixel p plus a threshold value t or all darker than the intensity of p minus threshold value t, then p is classified as point of interest [9]. Values of N and t can be chosen experimentally [8]. One of the most important features of this algorithm is its computational speed. We adapted this algorithm for our purposes by changing the radius of Bresenham circle from 3 to 4 for resolution from Full HD to UHD and to from 3 to 5 for images with resolution 4K and larger. [14]

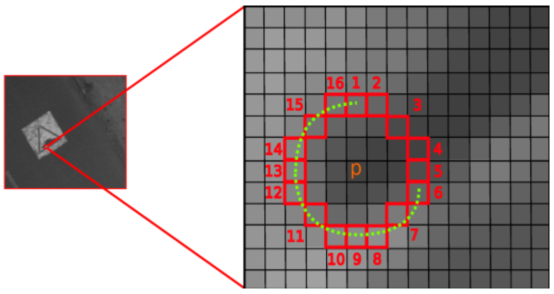


Fig. 1. Visualization of original FAST algorithm [9]

C. Convolutional Neural Network

A typical CNN consists of a single input, single output as well as multiple hidden layers. The hidden layers of a CNN are typically convolutional layers, pooling layers, fully connected layers and normalization layers [10].

The input layer defines the size of the input vectors/matrices of a convolutional neural network and in case of images contains the raw pixel values of the images. This layer also often performs data normalization by subtracting the mean image of the training set from every input image [11].

A convolutional layer consists of neurons that connect to the regions of the input images or the outputs of the preceding layer. A convolutional layer learns the features localized by these regions while scanning through an image. The size of these regions is defined using the filter size [12][15][16].

In the training phase the network computes a dot product of the weights with the input, and then adds a bias term. A set of weights that are applied to a region in the image is called a filter. The filter is then moved along the input image vertically and horizontally, repeating the same procedure for each region, that is, convolving the input. As the filter moves along the input, it uses the same set of weights and bias for the convolution, forming a feature map. The convolutional layer produces precisely as many feature maps, as the number of filters [11].

Convolutional and batch normalization layers are usually followed by a non-linear activation function such as a rectified linear unit (ReLU) performing a threshold operation to each element, where any input value less than zero is set to zero. The ReLU layer does not change the input size [12][13].

Batch normalization layers are used between convolutional layers and non-linearities such as ReLU layers to speed up network training and reduce the sensitivity to network initialization. The layer first normalizes the activations of each channel by subtracting the mini-batch mean and dividing by the mini-batch standard deviation [13].

Max and average-pooling layers follow the convolutional layers for down-sampling, hence, reducing the number of connections to the following layers (usually a fully connected layer). They do not perform any learning themselves but reduce the number of parameters to be learned in the following layers. They also help reduce overfitting. A max-pooling layer returns the maximum values of rectangular regions of its input [11].

The convolutional layers are followed by one or more fully connected layers. All neurons in a fully connected layer connect to all the neurons in the previous layer. This layer combines all the features (local information) learned by the previous layers across the image to identify the larger patterns [12][15][16].

For classification problems, a softmax layer and then a classification layer must follow the final fully connected layer. The output of the softmax layer consists of positive numbers that sum to one, which can then be used as classification probabilities by the classification layer which uses the probabilities returned by the softmax activation function to assign the input to one of the mutually exclusive classes and compute the loss [11][15][16].

III. Proposed Algorithm

The first step of our proposed algorithm is image pre-processing using color filters and cropping. This algorithm preforms a quick skin color detection and removal of all other colors. Only the largest skin color object in the image is preserved.

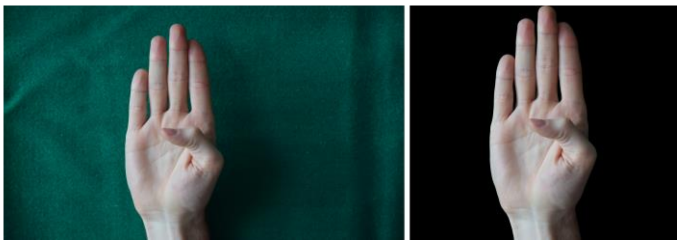


Fig. 2. Input image pre-processing

In the same step the hand is centered and cropped from all sides to create the smallest possible square. Cropping is accomplished by cropping continuous zero elements of the original matrix, until a new square matrix is formed. The square image is then convolved with discrete LoG filter. The size of the filter is dependent on the resolution of the input image. The filter size can be 11x11 for small image such as 512x512, 21x21 for images with resolution 1024x1024, 41x41 for 2048x2048 images and 81x81 for 4096x4096 or UHD/4K images. This filter can be scaled indefinitely with increasing resolution. Sequences captured by CMOS DSLR were used during the creation and testing of this algorithm. Their original resolution was 6000x4000 and after preprocessing they were cropped to 3800x3800 on average.

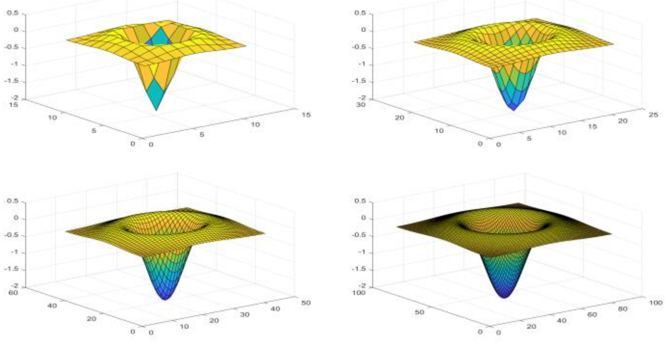


Fig. 3. Visualization of LoG filters with size 11x11, 21x21, 41x41 and 81x81

Resulting matrix obtained by the convolution of cropped ASL image with a LoG filter is visualized in figure 4.

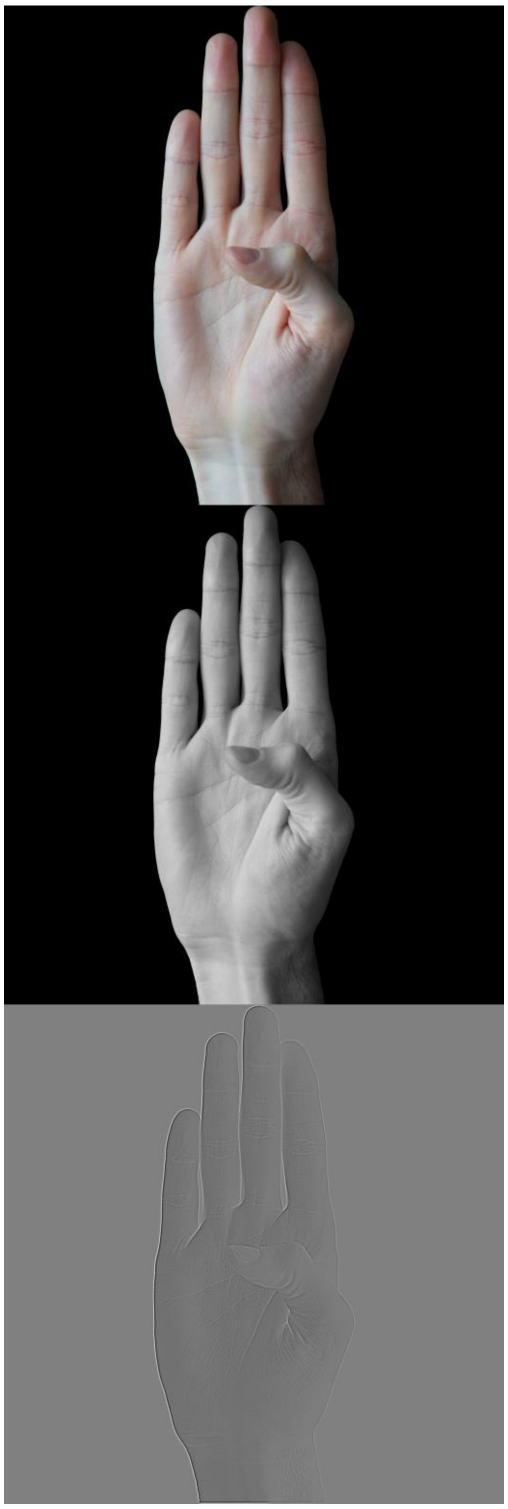


Fig. *4*. Application of LoG filter to hand image.

The result is a smoothed edge image that preserves edges and high frequency components of the original image. The image is converted to greyscale before the convolution in order to reduce the computational requirements in further steps. Such a conversion does not influence the ability to recognize the image.

The next step of our algorithm is the application of adaptive FAST algorithm. The size of Bresenham circle was set to 5 for the used input images (mostly square image with size 4000x4000). The resulting points of interest and thus feature components of feature vector are displayed in figure 5.

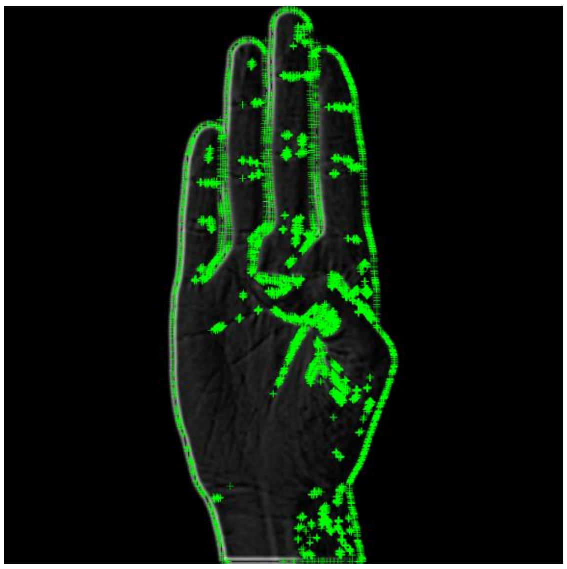


Fig. *5*. Position of FAST feaure point in image

Relative positions of these key-points is extracted from x, y position of key-points divided by the size of the analyzed image and saved in a feature vector. Figure 6 displays only the positions of these key-points extrapolated from the feature vector to a 512x512 matrix.

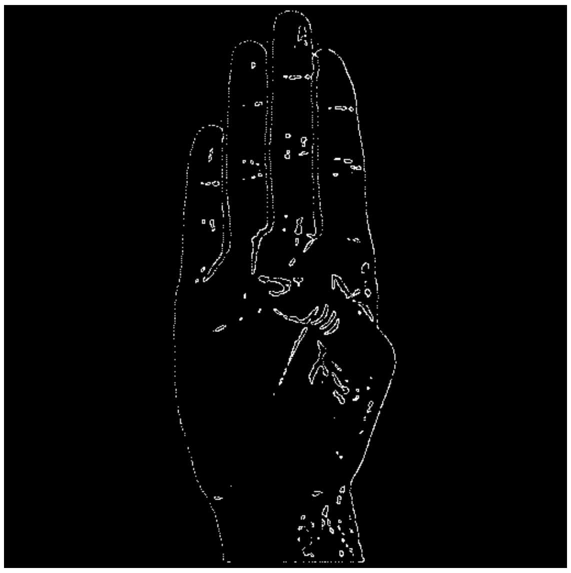
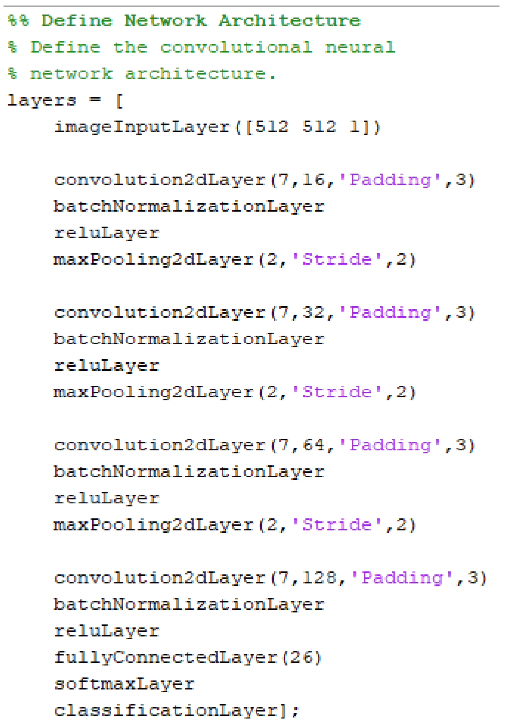


Fig. *6*. Key-points displayed in a 512x512 matrix

The last step is using the CNN for feature vector classification. The used network was trained using feature vectors obtained from other images of the same ASL alphabet signed by different people. For the purposes of testing our convolutional neural network was implemented in MATLAB. It consisted of 19 layers, configuration of which is displayed in the MATLAB capture (figure 7).



```
%% Define Network Architecture
% Define the convolutional neural
% network architecture.
layers = [
    imageInputLayer([512 512 1])

    convolution2dLayer(7,16,'Padding',3)
    batchNormalizationLayer
    reluLayer
    maxPooling2dLayer(2,'Stride',2)

    convolution2dLayer(7,32,'Padding',3)
    batchNormalizationLayer
    reluLayer
    maxPooling2dLayer(2,'Stride',2)

    convolution2dLayer(7,64,'Padding',3)
    batchNormalizationLayer
    reluLayer
    maxPooling2dLayer(2,'Stride',2)

    convolution2dLayer(7,128,'Padding',3)
    batchNormalizationLayer
    reluLayer
    fullyConnectedLayer(26)
    softmaxLayer
    classificationLayer];
```

Fig. *7*. MATLAB capture of CNN configuration

The chosen training method for our network was SGDM (Stochastic Gradient Descent with Momentum). The entire database consisted of 26 signs, each with 10 samples of the same sign. [15][16]

The number of samples was low due to low availability of ASL high resolution videos and photographs. All our database samples had to be created by volunteers. We had to increase the size of the database artificially to compensate for the low number of samples and differences in the shape of human hand.

The database images were manually rotated and distorted. They were all thickened and thinned from -5% to +5% with the step 1%. In this way we turned one database image into 11 images.

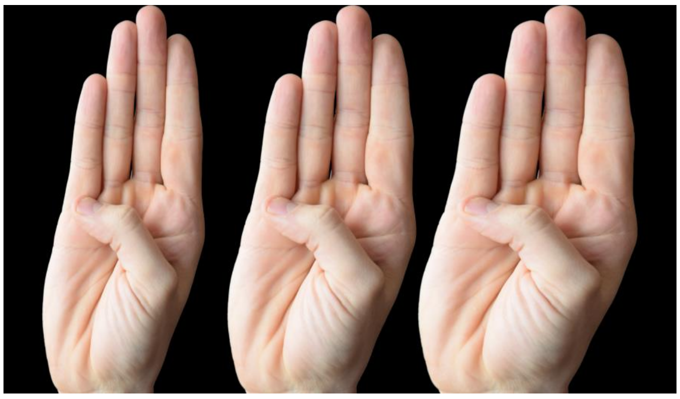


Fig. *8*. Thin, original and thick version of image.

Figure 8 shows what the -5% thinner and +5% thicker version of the hand looks like compared to the original.

It was also imperative to compensate for different possible rotations for each hand posture. The training data set was therefore extended with different rotations of each sign's original image.

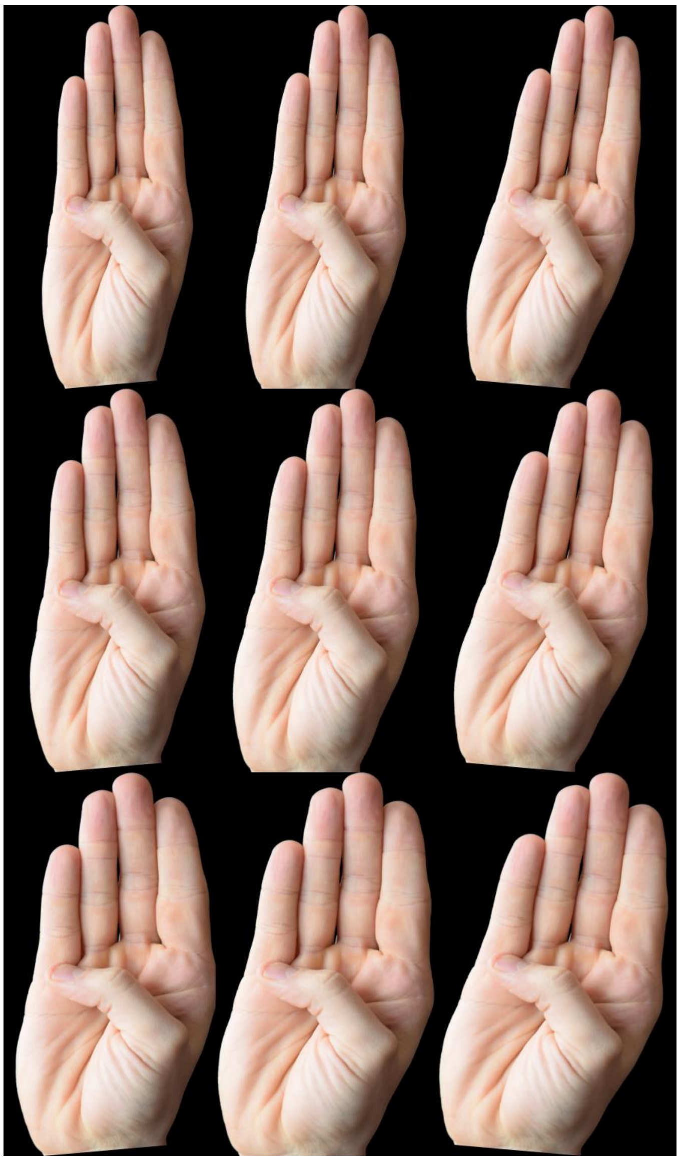


Fig. *9*. Rotated and reshaped of the same hand

Figure 9 displays different variants of the rotated hand image, rotated from -5° to the left to +5° to the right with a 1° step.

After both extensions were applied, 1 sample image in the training data set was transformed into 121 separate training images. All 10 samples were thus transformed into 1210 samples. Multiplied by 26 different signs, the final size of the available data counted 31460 images.

IV. Proposed Algorithm

Various configurations of convolutional networks were tested with the one in figure 7 providing the best results. The training process took 30 minutes on average.

9 sets of samples (9 x 121 images) for each sign were used for training and one set (121 images) for testing. After the testing was complete, the order of which signs were used for testing and training was changed. The best recognition rate reached was 96,79% and it was reached using trained neural network from figure 7. The average recognition rate for all trainings was 91.34%.

he recognition rate was influenced by many factors, most notably by the very small size of the training database. Most CNNs require training sets with hundreds of thousands of samples for each classification category to reach ideal recognition rates. Our database consisted of many variations of the same data. Based on the experience with many different applications and image databases, the potential recognition rate can be further improved to get close to 98% by massively increasing the training dataset [17].

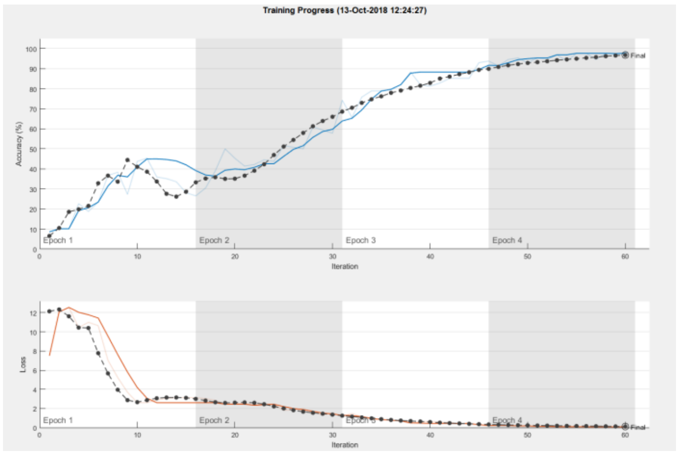
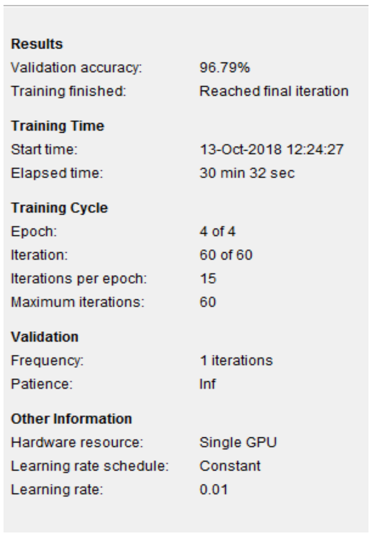


Fig. *10*. Visualization of CNN training and recognition process



Results

Validation accuracy:	96.79%
Training finished:	Reached final iteration

Training Time

Start time:	13-Oct-2018 12:24:27
Elapsed time:	30 min 32 sec

Training Cycle

Epoch:	4 of 4
Iteration:	60 of 60
Iterations per epoch:	15
Maximum iterations:	60

Validation

Frequency:	1 iterations
Patience:	Inf

Other Information

Hardware resource:	Single GPU
Learning rate schedule:	Constant
Learning rate:	0.01

Fig. *11*. Best results reached

Figures 10 and 11 show the training process in MATLAB. The training was performed on a GPU. The feature extraction process for each image took on average 0.432246 seconds and the classification took 0.005951268244576 seconds. This process allows for recognition of 2 images per second. The processing speed of MATLAB is quite low compared to native languages such as C. It is stated on MATLAB forums that the same optimized application implementation natively in C is roughly 10 times faster than in MATLAB (this is a very rough estimate and requires further verification). Taking this into consideration, the recognition rate of our algorithm could theoretically reach 20 sign language images per second, which is far greater speed than any individual person can sign per second. We believe that the algorithm has quite a large potential for various applications, considering that it was able to classify images with resolution 4000x6000 pixels.

References

[1] H. Chen, T. Ballal, M. Saad and T. Y. Al-Naffouri, "Angle-of-arrival-based gesture recognition using ultrasonic multi-frequency signals," 2017 25th European Signal Processing Conference (EUSIPCO), Kos, 2017, pp. 16-20.

[2] S. Routray, A. K. Ray and C. Mishra, "Analysis of various image feature extraction methods against noisy image: SIFT, SURF and HOG," 2017 Second International Conference on Electrical, Computer and Communication Technologies (ICECCT), Coimbatore, 2017, pp. 1-5.

[3] Q. B. Dang, V. P. Le, M. M. Luqman, M. Coustaty, C. D. Tran and J. M. Ogier, "Camera-based document image retrieval system using local features - comparing SRIF with LLAH, SIFT, SURF and ORB," 2015 13th International Conference on Document Analysis and Recognition (ICDAR), Tunis, 2015, pp. 1211-1215.

[4] K. O. Rodríguez and G. C. Chávez, "Finger Spelling Recognition from RGB-D Information Using Kernel Descriptor," 2013 XXVI Conference on Graphics, Patterns and Images, Arequipa, 2013, pp. 1-7.

[5] A. Anand, S. S. Tripathy and R. S. Kumar, "An improved edge detection using morphological Laplacian of Gaussian operator," 2015 2nd International Conference on Signal Processing and Integrated Networks (SPIN), Noida, 2015, pp. 532-536.

[6] A. Anand, S. S. Tripathy and R. S. Kumar, "An improved edge detection using morphological Laplacian of Gaussian operator," 2015 2nd International Conference on Signal Processing and Integrated Networks (SPIN), Noida, 2015, pp. 532-536.

[7] H. Kong, H. C. Akakin and S. E. Sarma, "A Generalized Laplacian of Gaussian Filter for Blob Detection and Its Applications," in IEEE Transactions on Cybernetics, vol. 43, no. 6, pp. 1719-1733, Dec. 2013.

[8] Y. Biadgie and K. A. Sohn, "Feature Detector Using Adaptive Accelerated Segment Test," 2014 International Conference on Information Science & Applications (ICISA), Seoul, 2014, pp. 1-4.

[9] L. Guo, J. Li, Y. Zhu and Z. Tang, "A novel Features from Accelerated Segment Test algorithm based on LBP on image matching," 2011 IEEE 3rd International Conference on Communication Software and Networks, Xi'an, 2011, pp. 355-358.

[10] D. Ravenscroft et al., "Learning feature extractors for AMD classification in OCT using convolutional neural networks," 2017 25th European Signal Processing Conference (EUSIPCO), Kos, 2017, pp. 51-55.

[11] H. Yang, C. Yuan, J. Xing and W. Hu, "SCNN: Sequential convolutional neural network for human action recognition in videos," 2017 IEEE International Conference on Image Processing (ICIP), Beijing, China, 2017, pp. 390-394.2017 IEEE International Conference on Image Processing (ICIP), Beijing, China, 2017, pp. 355-359.

[12] Q. Wang, H. Fan, Y. Cong and Y. Tang, "Large receptive field convolutional neural network for image super-resolution," 2017 IEEE International Conference on Image Processing (ICIP), Beijing, China, 2017, pp. 958-962.

[13] C. C. Hsu and C. W. Lin, "Unsupervised convolutional neural networks for large-scale image clustering," 2017 IEEE International Conference on Image Processing (ICIP), Beijing, China, 2017, pp. 390-394.

[14] Zhe Xu, Biao Min, Ray C.C. Cheung, “Key-point Detection based Fast CU Decision for HEVC Intra Encoding” in International Journal of Electronics and Telecommunications, 2018, Volume 64, No. 3, ISSN:2081-8491

[15] Sebastian Temich, Damian Grzechca, “Application of Neural Network for Testing selected specification parameters of Voltage-Controlled Oscillator” in International Journal of Electronics and Telecommunications, 2018, Volume 64, No. 2, ISSN:2081-8491.

[16] Y. Zhao and L. Wang, "The Application of Convolution Neural Networks in Sign Language Recognition," 2018 Ninth International Conference on Intelligent Control and Information Processing (ICICIP), Wanzhou, China, 2018, pp. 269-272.

[17] W. Fakhr, M. Kamel and M. I. Elmastry, "Probability of error, maximum mutual information, and size minimization of neural networks," *[Proceedings 1992] IJCNN International Joint Conference on Neural Networks*, Baltimore, MD, USA, 1992, pp. 901-906 vol.1