Gesture Recognition for American Sign Language Using Pytorch and Convolutional Neural Network

Devashsih Sethia, Pallavi Singh, and B. Mohapatra

Abstract Human–computer interaction (HCI) is the most prevalent topic of active research due to the demand for machine learning and computer vision. American Sign Language (ASL) is one of the most popular languages used by deaf and dumb people in the world. The deaf and dumb people use hand gestures to communicate. Hand gestures vary from person to person in shape, size, scale, and image quality. Hence, nonlinearity exists in this problem. In the area of image processing, there has been tremendous progress made recently, and it's proven that neural networks have numerous applications in interpreting sign language. The recognition of ASL in real-time motion is employed using an efficient artificial intelligence tool, and Convolutional Neural Network (CNN) has been proposed in this work. The dataset of 27,455 images of 25 English alphabets has been used to train and validate our model. The model is tested on 7172 images which were divided into many classes. The maximum validation accuracy of the model with enhanced data was found to be 99.8% which is better than many existing methods in real-time motion.

Keywords Sign language · CNN · Gesture recognition · Pytorch

1 Introduction

ASL (American Sign Language) is the most widely used sign language in the USA. Since deaf and dumb people's main limitation is communication, and they are unable to communicate via spoken languages. Sign language is their only means of communication. Deaf and dumb persons use their hands to express various gestures in order to communicate with others. Gestures are non-verbally communicated messages that are recognized by eyesight. Sign language is the nonverbal communication of the deaf and dumb. There are a variety of sign languages, including British, Indian, and American. Users of ASL may struggle to understand British sign language (BSL), and vice versa.

D. Sethia · P. Singh · B. Mohapatra (✉)
Department of Electrical, Electronics and Communication Engineering, Galgotias University, Gr. Noida 201310, India
e-mail: writetobm@gmail.com

A. J. Kulkarni et al. (eds.), *Intelligent Systems and Applications*, Lecture Notes in Electrical Engineering 959, https://doi.org/10.1007/978-981-19-6581-4_24

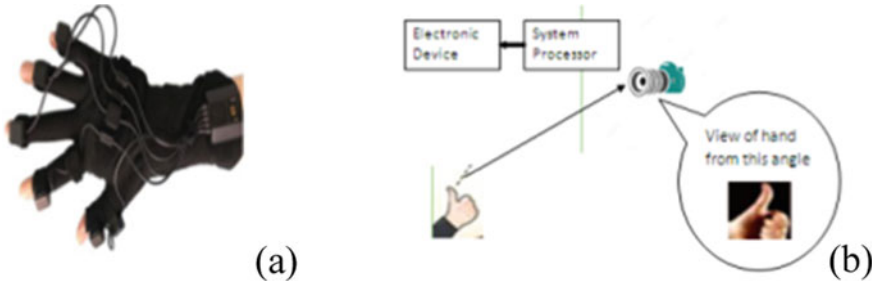

Fig. 1 **a** Sensor-based technique and **b** vision-based techniques

Numerous methods for evaluating gestures/sign language have been developed. First of all, there is a technique that requires users to wear sensors (input devices) such as gloves and hand straps shown in Fig. 1a. These strategies offer the advantage of not being distracted by various backgrounds when identifying motions. But they come at the cost of lack of natural resources, size, and expense. Secondly, camera-vision strategies use imaging tools such as cameras or kinetic sensors to capture data based on the perception of how a person perceives in their environment shown in Fig. 1b. The placement of cameras, hand visibility and the way it is segregated in the image, the effectiveness of the extricate characteristics, and the classification methods all contribute to the efficiency of these systems [1].

The biggest problem is finding a way to train a computer to detect the motion of hands. The adjustment of the fingers and the form of the hands vary according to the motion of the hands. As a result, uncertainty is one of the aspects of hand gestures that should be considered. The information of the data conveyed by the images can be used to accomplish the uncertainty in hand gestures. This method involves combining two tasks: feature extraction and classification. Traditional image processing algorithms-based hand gesture recognition was not generally used in HCI due to its limited low accuracy and complexity of the algorithm. An account of development in gesture detection established on machine learning has recently progressed quickly in HCI [2, 3].

Traditional pattern recognition techniques are unable to analyze intrinsic information of the data in its original form [4]. Subsequently, it results in extracting features from original configuration which consumes time, and it is not automated. For classification, CNN's are used which are a type of deep neural network. Deep neural networks extricate features on the go [5] and employ completely linked layers for classification by using CNN. Furthermore, CNN comprehends the complicacy of the visual images and uncertainty between the images. Therefore, using a CNN-based feed forward network approach is used to solve the problem. This is the most well-known method of detection using several neural networks.

Despite the fact that the hand motions are static, they are recognized in real time in this proposed model. However, the extraction procedure is time-consuming, and it is unlikely that all available features will be extracted. Since, the neural network is highly recognized, the most infamous method which is CNN is useful to extract

features out of organized information. Thus, there is a retraction toward automated feature extraction and neural network, leading to deep neural network or CNN, emerging as a solution.

The following sections of the paper are described as follows: Sect. 2 presents literature survey and related work, proposed methodology has been presented in Sect. 3. Section 4 describes results and discussion, whereas conclusion and future works have been discussed in Sect. 5.

2 Literature Review

Several significant studies in the domain of hand gesture recognition are discussed as it has been a burning topic of numerous studies. An ANN-based gesture detection system has been developed [6]. In this research work, the images were classified in this manner depending on skin tones. Changes in pixel through cross-sections, border, and scalar descriptions like aspect ratio and edge ratio were chosen as ANN features. After those feature vectors were established, they were sent to the ANN for training. The author claimed accuracy of the system estimated to be approximately 98%. A statistical technique for detecting gestures based on Haar-like traits has been proposed [7]. The AdaBoost technique was utilized to boost the classifier in this system. A model for real-time hand gesture identification was presented [8]. These gestures are collected using a commercial surface sensor called the Myo wristband, which sends data to the computer through Bluetooth. Signal acquisition, preprocessing, feature extraction, classification, and postprocessing are the five stages of the proposed model. They employed the KNN rule in conjunction with the dynamic time warping technique for the classification stage. The claimed model's accuracy was 86%. A novel approach to identify hand gestures in a complex environment based on Single-Shot Multibox Detector (SSD) which is a deep learning algorithm which uses a neural network with 19 layers [9]. The image pyramid method is used to recognize gestures in this system. To recognize both large and small hand gestures, the algorithm crops the image into blocks. The model is tested using the hand gestures of four characters in three different complex backdrops. The system's response time is below 20 ms, indicating excellent real-time performance. The least accuracy is over 93.8%, with a high accuracy of 99.2%. The mechanism, however, is only regulated for four characters: "w", "o", "r" and "k".

A 3D CNN-based method for recognizing hand gestures has been proposed [10]. The identification in this method was challenging as capturing the gestures images with different depth and a light intensity skewed the results. They acquired 77.5% accuracy on the VIVA challenge dataset by utilizing a data augmentation approach. An approach to recognize gestures using CNN that is resilient under five invariants has been developed by Flores et al. [11]. These invariants include illumination, noise, scale, rotation, background, and translation. The dataset used was Sign Language of Peru (LSP). Architecture of two CNN models was constructed which is related to static signs and achieved 95.37% accuracy for the first CNN and 96.20% accuracy

for the second CNN. Two unique CNN models capable of recognizing 24 static ASL signs were developed. They separate the ASL signs into two datasets. One into color images and the other into a combination of color and depth images. The author claimed an accuracy for the two datasets which is 86.52% and 85.88%, respectively. The results of these two sets of datasets were compared utilizing transfer learning with pre-trained models like VGG-19, VGG 16, and others [12]. To recognize sign alphabets, a system was suggested that used both deep learning models as well as image processing techniques. Using YCbCr, the images are segmented. The characteristics from handcrafted methods and a deep learning network termed VGG-19 are combined. They are delivered to the Support Vector Machine classifier to classify the signs using the serial fusion method. The accuracy achieved by the proposed system was roughly claimed to be 98.44% [13].

3 Proposed System and Methodology

In this approach, the majority of this assignment's usage has been handled through transfer learning; however, our network has been created from scratch. To analyze the image and train the model, our model uses CNN architecture. There are many CNN layers which are accompanied further by a pooling layer, ReLU layer, batch normalization layer, and dropout layer in the architecture. The architecture is composed of the following layers: Input, two convolution 2D layer, two batch normalization layer, two activation function (ReLU), two pooling 2D layer, two dropout layer which is further followed by two fully connected layers, an activation function (ReLU) and a LogSoftmax layer as output.

The collection of dataset and CNN which were employed with the feed forward network are described in this section. Figure 2 depicts the conventional method of the process flowchart. The method entails gathering data, initializing the preprocessing method, the CNN with feed forward network is composed, construction of model is done by training and tuning the model, and later by testing it.

Step-1: Data Collection

The gesture recognition technique contains three main components. According to the previous [7] research, the gesture data collection module, the feature extraction module, and the gesture classification module are used. The gesture classification module employs an appropriate classifier to classify distinct movements. For additional step recognition, researchers propose definite characteristics extrication methodology, for instance, range and inclination from the hand's terminus [14] and

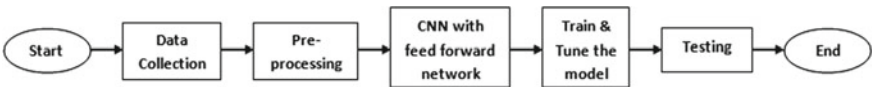

Fig. 2 Conventional flowchart process

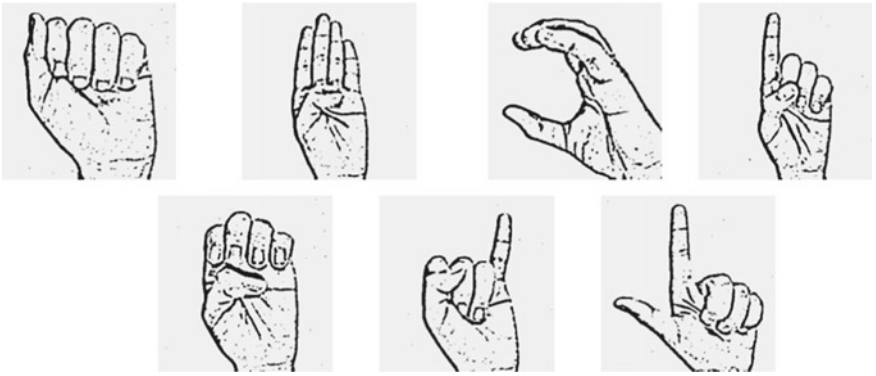

Fig. 3 Sample dataset

orientation histogram [14]. Furthermore, we've chosen 25 static gestures as the recognizable data. The dataset format is based on the famous MNIST dataset, which is a collection of English letters (A–Z). For each alphabet A–Z, each training and test case represents a label (0–25) as a one-to-one map. For instance, there are no test cases for the 9th letter which is J and the 25th letter which is Z. This is because of hand movements made while doing this gesture. There are 24 alphabet signs that have been taken for the process. As the alphabet J and Z requires movement, it becomes hard for the system to recognize that in real time. Therefore, we have 24 alphabets as data and one empty set sample of which is shown in Fig. 3. Hence, there are 27,455 images utilized for training the model and 7172 images utilized for testing.

To gather the images of the hand gestures, a computer camera was employed. The sign language gestures were made in front of the computer camera in an effort to train and validate the model. The input images are believed to contain exactly one hand, with the palm facing the camera. However, the identification procedure of the images would be more effective and less complex, if the surrounding of the image is not complicated and the contrast is high. Therefore, simple backgrounds of the images and more uniformity were assumed to be considered.

Step-2: Preprocessing

A basic preprocessing method was conducted with the collected data. This was done aiming to achieve better efficiency and less computational complexity. Subsequently, the frame size is initially set to 20 × 20 mm. The backdrop of the images is then eliminated using OpenCV's background removal method [15, 16]. Background subtraction aids in obtaining relatively quick and preliminary identifications of the items in the video stream, which can then be handled more delicately. After removal of the background, as the original image is preprocessed only the remnant of the hand is left. Furthermore, the original image is converted into the monochromatic image as it is gray scaled. It is easier for CNN to learn and it takes less time as well [17–20]. The image is also modified to fit within the desired image area. The photographs were then downsized to fit into a 20 × 20 frame and submitted to CNN. The image is

thresholded after it has been reshaped and resized. The assignment of pixel values in respect to the threshold value is shown in the image. Furthermore, each pixel value is compared to the threshold value while thresholding shown in Fig. 4.

Step-3: CNN with feed forward network

CNN is particularly effective in the image identification process. As a result, it's also suitable for real-time processing. The CNN has three dimensions: width, height, and depth. A CNN is a multi-layer neural network in which each layer transfers activation amounts to another layer via a function. Convolution2D Layer, Rectified Linear Unit (ReLU) Layer, and MaxPooling2DLayer are used to extract features from data before it is transformed to the fully connected layer, LogSoftmax layer, and classification output layer. It is classified in Fig. 5.

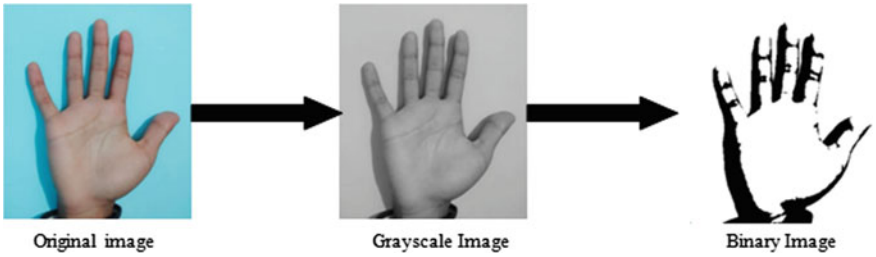

Fig. 4 Steps of preprocessing

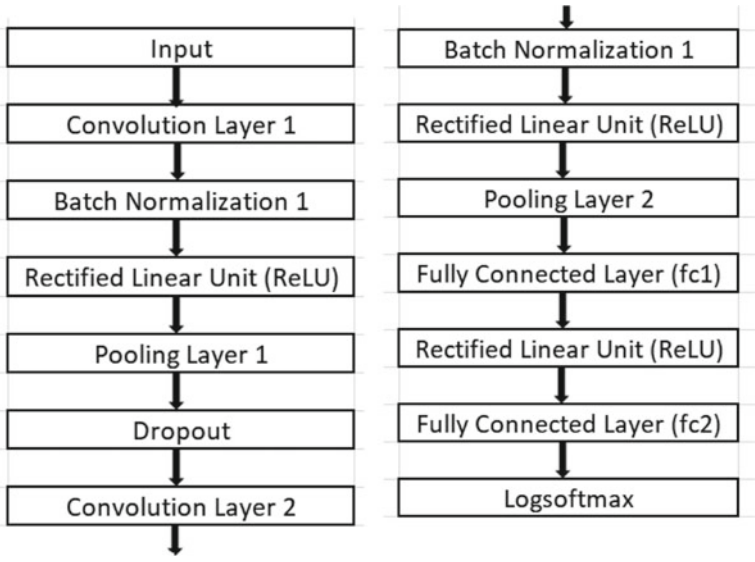

Fig. 5 Block diagram of the system

The two-dimensional Convolution2D Layer is used as the activation function. The kernel size of this layer is 5 × 5. It is necessary to give the input size and the stride has been set to default value. The input shape of the layer is 80 × 80 × 1. Thus, hinting that a monochromatic image of 80 × 80 pixels will be sent to the CNN network. The next layer is batch normalization 2D layer which is used for accelerating deep network training by reducing internal covariate shift. It's a technique for standardizing network inputs that can be applied to the activations of a previous layer or directly to the inputs. For the uncertainty issue, ReLU was introduced [15]. This is an activation function which surpasses all other functions. Thereafter, a pooling two-dimensional layer (MaxPooling2DLayer) with a 2 × 2 pool size that takes the maximum value was added. It just selects the utmost significant information; this layer assists the network's understanding of the images which is followed by dropout layer as it prevents overfitting.

The next layer is another convolution 2D layer followed by batch normalization 2D layer. ReLU which is an activation function layer comes next. Finally, it goes to the max pooling two-dimensional layer with the pool size of 2 × 2 and so as to stop from overfitting, a dropout layer is employed. Furthermore, the next layer comprises a feed forward network which is fully connected having several nodes. ReLU is used as the activation function in this layer. Moreover, the linear module included creates a single layer feed forward network with 1280 input features and 250 output. This linear layer is capable of learning an average rate of correlation between the output and the input. Another feedforward linear layer with 250 input and 25 output is used.

The output layer uses one dimensional LogSoftmax function. The output of this layer is a logarithmic probability. In addition, on utilizing Pytorch open source machine learning library, this system is composed. Eventually, to retain the trail of the analysis procedure using the proposed CNN configured model the criterion of deprivation and precision were described.

Step-4: Train and Tune the model

A properly constructed CNN with correct training code should be able to memorize the answers to a set of images data. So, in order to train the neural network model, OpenCV is used for compilation and Pytorch as an open source machine learning library is employed. Also, one dimension LogSoftmax is used which is a software based activation function.

To train the model, some hyper-parameters were included in order to tune the model. Therefore, some transformations were done like re-sizing, scaling up, and down the image, snipping, breadth, and peak shifting on the dataset. The listed hyper-parameters were tuned. After training and tuning the model, testing of the system is required. Our model was trained to achieve good performance, mentioned in Sect. 4. The model is trained using the expanded dataset and tested as well. Furthermore, to increase the number of data, data enhancement is done by applying flip, zoom, shear, etc. Figure 6 shows the effect of enhanced data, as it depicts the flipping of a sign language. Data enhancement is mainly a technique to increase the number of variables where those variables are the data.

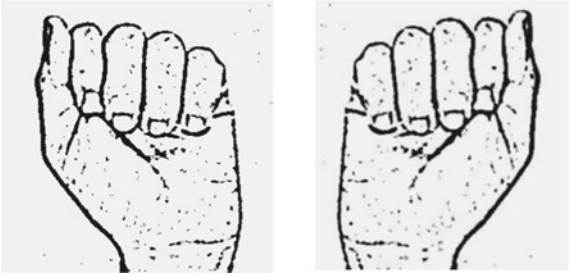

Fig. 6 Flipped images of same gestures

4 Results

Python was utilized as the programming language to implement the system. Customarily, code was written and run by utilizing OpenCV which is a computer vision library. CNN classifier has been created using Pytorch, an open-source machine learning library. For array operations, NumPy was employed. Imutils is used for doing simple image processing tasks including translation, rotation, and scaling. We used a CNN with a feed forward network to experiment with recognition of ASL in this study. We acquired, labeled, and trained the dataset used. Furthermore, the intended work was to implement hand motions in real time on low-power computing systems.

The American Sign Language detection system was tested on characters "A", "B", "C", "D", "E", "F", "G", "H", "I", "K", "L", "M", "N", "O", "P", "Q", "R", "S", "T", "U", "V", "X" and "Y" which showed good performance. The results are given in Table 1. The model achieved a minimum accuracy of 96.71% for letter "O" and maximum accuracy of 99.88% for letter "L" in real-time motion. The results of letters "l" and "O" are shown in Fig. 7. In both phases, when we trained and tested the model for ASL recognition indicated that the second phase of the model provides higher accuracy then all other well-known algorithms. The enhancement of the expanded dataset of training samples is large which is 27,455 total samples. The Sign languages that show markable results achieved accuracy of above 99%. The letters achieving the high level predictions were characters "Y", "P", "W", "U", "D", "R", "V", "X", "G", "B" and "C". Meanwhile, there are letters with accuracy of above 98% and below 99% and some letters that have accuracy of below 98% and above 97% as given in Table 1.

The training method of the model has shown that the time taken in the training phase of the algorithm was average compared to other sturdy algorithms. The shortest recorded time for the observation was 200 ms in real-time motion which is a remarkable result. The mean classification time was about, although, in real-time the accuracy did differ but when the hand is stable the output is more accurate and precise. The bounding box is 20 $\times$ 20 mm which is small but it eliminates all the disturbances and focuses on the hand gestures. Sample of results has been shown in Fig. 8.

Table 1 Representing accuracy of all characters

Letter	Accuracy (%)	Letter	Accuracy (%)	Letter	Accuracy (%)
A	98.45	I	98.96	R	99.30
B	99.01	K	97.63	S	98.51
C	99.00	L	99.88	T	97.41
D	99.34	M	98.01	U	99.37
E	99.36	N	98.47	V	99.16
F	98.77	O	96.71	W	99.57
G	99.07	P	99.60	X	99.13
H	98.21	Q	97.09	Y	99.62

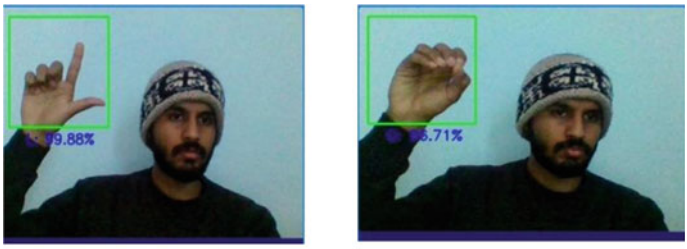

Fig. 7 **a** Maximum accuracy of letter "L" and **b** minimum accuracy of letter "O"

5 Conclusions and Future Work

We used both traditional image processing techniques and a deep learning model to recognize sign alphabets in this suggested system. We employed both handcrafted features like ASL and a deep learning network called CNN in this case. We conclude that CNN is a well-known component based procedure, and real-time gesture recognition has a substantial impact on deep learning as a result of our research. The proposed approach achieved an accuracy of 99.88%. As a result, more precise gesture recognition is possible. This system is limited to detect sign languages and the 24 alphabets. To enhance this, gestures of numerous signs shall be included. Sign language identification using both hands is also unachievable with this system. Hence, future work could be recognition of gestures with both hands. Moreover, the recognition should not just be limited to alphabets and numbers. Rather, recognition of sentences, speech, and much more should be encouraged.

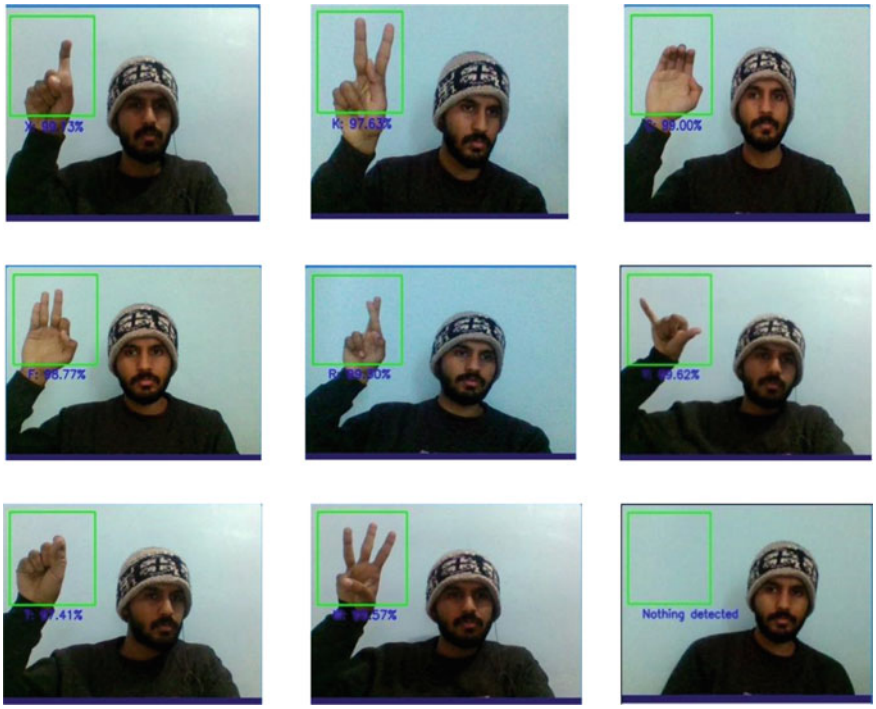

Fig. 8 Sample results

References

1. Parvathy P, Subramaniam K, Venkatesan GKDP, Karthikaikumar P, Varghese J, Jayasankar T (2020) Development of hand gesture recognition system using machine learning. J Ambient Intell Hum Comput 12:6793–6800. https://doi.org/10.1007/s12652-020-02314-2
2. Dongarra J, Gates M, Kurzak J, Luszczek P, Tsai YM (2018) Autotuning numerical dense linear algebra for batched computation with GPU hardware accelerators. Proc IEEE 106:2040–2055. https://doi.org/10.1109/JPROC.2018.2868961
3. Morchid M (2018) Parsimonious memory unit for recurrent neural networks with application to natural language processing. Neurocomputing 314:48–64. https://doi.org/10.1016/j.neucom.2018.05.081
4. LeCun Y, Bengio Y, H Ginton (2015) Deep learning. Nature 521:436. https://doi.org/10.1038/nature14539
5. LeCun Y, Haffner P, Bottou L, Bengio Y (1999) Object recognition with gradient-based learning. In: Shape, contour and grouping in computer vision, pp 319–345. https://doi.org/10.1007/3-540-46805-6_19
6. Nguyen T-N, Huynh H-H, Meunier J (2013) Static hand gesture recognition using artificial neural network. J Image Graph 1:34–38. https://doi.org/10.12720/joig.1.1.34-38
7. Chen Q, Georganas ND, Petriu EM (2008) Hand gesture recognition using Haar-like features and a stochastic context-free grammar. IEEE Trans Instrum Meas 57:1562–1571. https://doi.org/10.1109/TIM.2008.922070
8. Benalcázar ME, Jaramillo AG, Zea JA, Páez A, Andaluz VH (2017) Hand gesture recognition using machine learning and the Myo armband. In: Proceedings of the 25th European signal

processing conference (EUSIPCO), 28 Aug–2 Sept 2017, pp 1040–1044. https://doi.org/10.23919/EUSIPCO.2017.8081366
9. Liu P, Li X, Cui H, Li S, Yuan Y (2019) Hand gesture recognition based on single-shot multibox detector deep learning, 30 Dec 2019, pp 1–7. https://doi.org/10.1155/2019/3410348
10. Molchanov P, Gupta S, Kim K, Kautz J (2015) Hand gesture recognition with 3d convolutional neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition workshops, 7–12 June 2015, pp 1–7. https://doi.org/10.1109/CVPRW.2015.7301342
11. Flores CJL, Cutipa AG, Enciso RL Application of convolutional neural networks for static hand gestures recognition under different invariant features. In: Proceedings of the 2017 IEEE XXIV international conference on electronics, electrical engineering and computing (INTERCON), Cusco, Peru, 15–18 Aug 2017, pp 1–4. https://doi.org/10.1109/INTERCON.2017.8079727
12. Paul P, Bhuiya M, Ullah M, Saqib MN, Mohammed N, Momen S (2019) A modern approach for sign language interpretation using convolutional neural network. In: Proceedings of the 16th Pacific rim international conference on artificial intelligence, Cuvu, Yanuca Island, Fiji, 26–30 Aug 2019, Part III, pp 431–444. https://doi.org/10.1007/978-3-030-29894-4_35
13. Rajan RG, Judith Leo M (2020) American sign language alphabets recognition using hand crafted and deep learning features. In: Proceedings of the 2020 international conference on inventive computation technologies (ICICT), 26–28 Feb 2020, pp 430–434. https://doi.org/10.1109/ICICT48043.2020.9112481
14. Mendoza-Garcia R, Landa-Hurtado L, Mamani-Macaya F, Valenzuela-Coloma H, Fuentes-Maya M (2014) Kinect-based trajectory teaching for industrial robots. In: Proceedings of the Pan-American congress of applied mechanics, Santiago, Chile, March 2014
15. Zivkovic Z (2004) Improved adaptive Gaussian mixture model for background subtraction. In: Proceedings of the 17th international conference on pattern recognition, 26 Aug 2004, pp 28–31. https://doi.org/10.1109/ICPR.2004.1333992
16. Zivkovic Z, Van Der Heijden F (2006) Efficient adaptive density estimation per image pixel for the task of background subtraction. Pattern Recogn Lett 773–780. https://doi.org/10.1016/j.patrec.2005.11.005
17. Grundland M, Dodgson NA (2007) Decolorize: fast, contrast enhancing, color to grayscale conversion. Pattern Recogn 2891–2896. https://doi.org/10.1016/j.patcog.2006.11.003
18. Singha J, Laskar RH (2015) Self co-articulation detection and trajectory guided recognition for dynamic hand gestures. IET Comput Vis 10:143–152. https://doi.org/10.1049/iet-cvi.2014.0432
19. Singha J, Laskar RH (2015) ANN-based hand gesture recognition using self co-articulated set of features. IETE J Res 61:597–608. https://doi.org/10.1080/03772063.2015.1054900
20. Singha J, Laskar RH (2016) Recognition of global hand gestures using self co-articulation information and classifier fusion. J Multimodal User Interfaces 10:77–93. https://doi.org/10.1007/s12193-016-0212-0