

# American Sign Language Gesture Recognition by Computer Camera

Nguyen Hoang Tuan<sup>1</sup>, Ly Trung Kien<sup>1</sup>, Dang Thanh Tin<sup>1</sup>

<sup>1</sup> Faculty of Electrical and Electronics Engineering,  
Ho Chi Minh City University of Technology, Ho Chi Minh City, Vietnam  
dttin@hcmut.edu.vn

**Abstract.** Nowadays, there are a lot of methods available for their communication for example: sign language, brain wave analysis, writing word, etc. The most popular and easiest to use is sign language. However, sign language is not popular for everyone especially ordinary people. So developing sign language system is very important, we aim to create a system which can help dumb, deaf and normal people who will be able to communicate easily with others. We focus on creating a vision based system to identify sign language gestures from the computer camera. The cause for selecting this system based on vision relates to the fact that it provides a simpler and more convenient way of communication between a human and a computer. In this project, we use Keras - Deep Learning API TensorFlow through python programming language and OpenCV library that can detect the sharp and movement of human's hand. By extracting featured images and comparing these with the available data which are trained before, our model will translate to readable languages.

**Keywords:** Keras, Sign language, Convolutional Neural Network(CNN).

## 1 Introduction

Human communicate with others by many ways but there are 2 basic ways: speech and gesture. According to a survey, about 15% of the world's population gets with some form of disability. Most of them are dumb or deaf so that they have a way of communication different from normal people. Disabled people can only use motions of any body language to express their idea. So that motion of any body parts are concentrated and builded a language system: sign language. Thanks to it, the communication of disabled people becomes easily but the problem of sign language is difficulty when normal people do not understand it. In our project, we are using image processing and computer vision to recognize and understand the human's gestures. It acts as an interpreter between computer and human. This could provide solution to human to interact naturally with the computers without any physical contact of the mechanical devices.

In the deaf and dumb community, normally sign language is used by everyone but this is the only way of communication for deaf and dumb community. Moreover, sign language are considered as the second language of the disabled community. Depending

on the geographical location and characteristics of the native language, the terminology is divided in many different ways such as: one hand or two hand. There are to form of sign language such as: isolated sign language and continuous sign language. Isolated sign language consists of single gesture having single word while continuous ISL or Continuous Sign language is a sequence of gestures that generate a meaningful sentence [10]. In this paper, we based on isolated ASL gesture recognition technique.

Sign language is a visual language and consists of 3 major components:

**Table 1.** Table captions should be placed above the tables.

Fingerspelling	Word level sign vocabulary	Non-manual features
Used to spell words letter by letter	Used for the majority of communication.	Facial expressions, tongue, mouth, and body position.

## 2 Related works

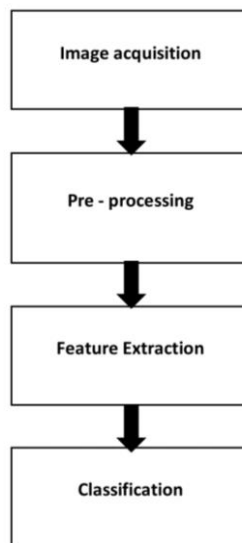
In the past, there were so many methods to support people with disabilities communicating with normal people. The biggest prevention nowadays are the complexity of sign language gesture. So in the early year, thank to the development of sensor technology, we had the wearable sensor or sensor glove which could recognize the movement of hand to translate from sign languages to literacy one with the very high accuracy in the short period [10]. In [1], we can see the model of Custom Glove having the highest accuracy with 99.75% when they practiced with 40 ASL signs. In this methods, they used the glove combined Ascension Technologies MotionStar 3D tracking system for speed reasons so that they can modulate the gesture of people in 3D space. Thank to it, they achieved the highest accuracy.

In [2], another methods used Microsoft Kinect RGB-D camera captured data. After that Convolutional Neural Networks (CNNs) [5] were utilised for training. Through the image processing, data was utilised to identify the sign language. It resulted in giving 97.71% accuracy with ASL and 98.81% with ISL. Although the accuracy of this model was lower than the wearable sensor, the potential of Kinect was better than this one. The research identified that the image processing is the future method of gesture recognition.

In fact, most of the current models are difficult to apply in practice because of the complexity and the operating costs when we considered to another model for example [8]. To achieve the high accuracy, we have to use a lot of devices, tools and modern technology. This point created an overwhelming problem when we apply it in practice. So the requirement of the simple system which close to users is extremely necessary. Our application is built up from existed source “Sign Language Recognition Using Hand Gestures” by Shadab Shaikh [3] using the computer camera. Building this app is based on the principles of gesture identification as letters, but in extension we take a sequence of frames of a complete action to form words.

### 3 System specification

In almost computer vision method, the input device is the images of hands or fingers. These methods require only a camera without the use of any extra devices. These systems extracted the biological characteristic by describing artificial vision systems that are implemented in software and/or hardware. This approach created a challenging problem as these systems need to be background invariant, lighting insensitive, person and camera independent to achieve real time performance. Moreover, such systems have to be enhanced to addapt to requirements, including accuracy and robustness.



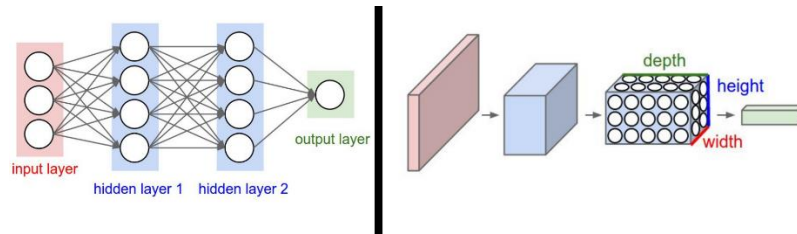
**Fig. 1.** Block Diagram of vision based recognition system.

Almost vision method is based on the way human get information in their environment but it is not easy to perform it in devices. a lot of different approaches have been verified so far. One of them is to build a three dimensional model of the human gesture. The system is compared with images of the gesture by one or more cameras, and parameters responses to palm direction and joint angles are estimated. Gesture classification is created by these parameters. Second one to get the picture using a camera then extract some feature and those features are input in a classification algorithm for classification.

## 4 Algorithms

### 4.1 Convolutional Neural Network (CNN)

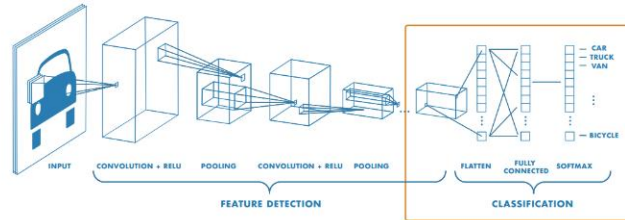
The convolutional neural network has a different architecture than the normal neural network. The normal neural network converts input through a series of hidden floors. Each floor is a set of neurons and fully-connected floors with neurons on the previous floor. And on the last floor will be the result layer that represents the prediction of the network. First, the convolutional neural network is divided into 3 dimensions: wide, high, and deep [7]. Next, the neurons in the network are not completely connected to the entire next neuron, but only to a small region. Finally, a minimalist output layer is vectorized to the probability value.



**Fig. 2.** Left: A regular 3-layer Neural Network. Right: A ConvNet puts its neurons in three dimensions [5].

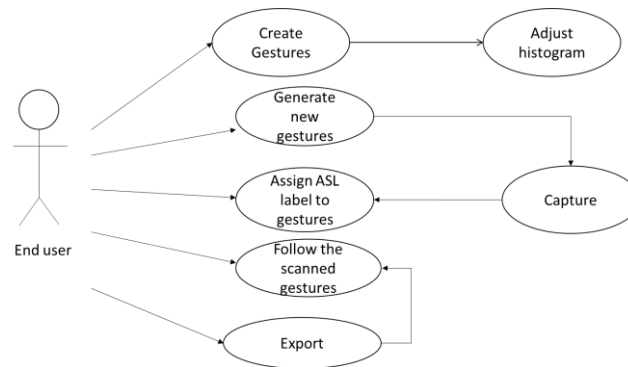
These layers choose one of three types of operations on the data: convolution, pooling, or rectified linear unit (ReLU). CNN system puts the input pictures into convolutional filters, each of which kicks certain characteristics from the pictures. Synthesis simplifies the output by using performing nonlinear downsampling, decreasing the number of parameters that the network must learn. Rectified linear unit (ReLU) allows for speed up and more amount training by mapping negative values to zero and maintaining positive values. These three functions are returned over hundreds or thousands of layers, with each layer learning to detect different characteristics.

**Classification Layers** After feature detection, the architecture of a CNN [5] shifts to classification. The next-to-last layer is a fully interacted layer that outputs a vector of  $K$  dimensions where  $K$  is the number of layers that the network will be able to anticipate. This vector includes of the probabilities for each class of any image being classified. The final layer of the CNN architecture [5] uses a softmax function to provide the classification output.



**Fig. 3.** . Architecture of a CNN [6].

#### 4.2 Use case diagram.



**Fig. 4.** Use Case Diagram for Sign Language Recognition Using Hand Gestures.

In our model, the users have 3 abilities: adding gestures, regconizing gestures and observing or equalizing manual parameters of training models. Users can create the gestures added our model through the user display. After that, the information will be trained the system by CNN [5].

#### 4.3 Module in the system.

**Data Pre-Processing** – In this application, based on the object detected in front of the camera its binary pictures is being common. The key object will be overlayed with solid white and background will be overlayed with solid black. Based on the pixel's regions, their numerical value in range of either 0 or 1 is being given to next process for modules.

**Scan Single Gesture** – A gesture scanner will be performed in front of the end user where the user will have to perform a sign language gesture. Based on Pre-Processed module output, a user shall be able to see associated label assigned for each hand gestures, based on the predefined American Sign Language (ASL) standard inside the output window screen [10].










Create gesture – A user will give an expected gesture as an input to the method with the text box at the bottom of the display. These customize gestures will then be saved for future aims and will be found in the next time.

Formation of a sentence – A user can choose a delimiter and until that delimiter is added every scanner gesture character will be inserted with the previous outcome forming a stream of meaningful words and sentences.

Exporting – A user can export the outputs of the scanner character into an textual file format.

#### 4.4 Training process.

To build the data set as a model and training we choose Vietnamese sign language. We opted out of the Ho Chi Minh Sign Language Dictionary for 12 words of sign language[9]. Each word in the archive is recorded with a total of about 2200 samples. These gestures are captured using the camera.

I	
YOU	
HELP	
PLAY	
CLEAN	
HOUSE	
SOCCER	
BORING	
AM	
OK	

**Fig. 5.** Some word of the training model.

After that, the best 2000 samples were selected. All of these images are extracted from the full animation of gestures, from the first frame to the last frame. The recorded sample will be processed to remove the background and keep the hand grayscale image.

For more detail, we restricted an area of the image captured by the camera, cut out that part and processed the color through the OpenVC library with “cvtColor” and “bitwise\_and”.

To start, we learn the training principle and we choose to convert a gesture sequence into multiple split frames and learn them in an orderly manner. Learning the frames broken down from an action will help maximize the machine's processing / learning capabilities and to learn these frames we choose the CNN [5] training method through Keras [4].

To decrease the amount of parameters we only need to learn the weights of the filter (which usually is a lot smaller than the input image) instead of learning the weights connecting each input pixel.

CNN method [5] is one of advanced deep learning paradigms. Especially, CNN [5] is widely used for solving object detection of images. The CNN [5] is divided into 3 dimensions: wide, high and deep. In the network, the neurons are not fully connected to the entire next neuron but they still have a relationship. Thus, an output layer is minimized to the vectors in those 3 dimensions.

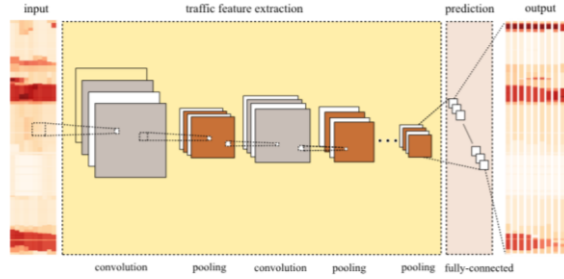
**Cnn\_model.py will train our data. Here are 4 steps pre-train (create a completed CNN).**

*Step 1.* Initialing sequentially the images, it's encoded by classifier.add and Convolution2D with 2 values 0 (black) and 1 (white) into the size  $64 \times 64 \times 3$ , convolution mean every element is multiplied with the matrix  $3 \times 3$  and sum up into 1 cell in new matrix that we call convolved feature. The convolved feature has the size  $64 \times 3 \times 32$ . The neurons from the Neural Network chapter remain unchanged: They still compute a dot product of their weights with the input followed by a non-linearity, but their connectivity is now restricted to be local spatially. Finally, we have 3 convotional layers with the recognition feature.

*Step 2.* Here we use Maxpooling method with Maxpolling2D. The pooling layer is used right after the convulational layer to simplify output information to reduce the number of neurons. The pooling procedure is applied as max-pooling, which selects the maximum value in the  $2 \times 2$  input area. Thus, through the Max Pooling layer, the number of neurons is halved. So every time we add a convolution layer, we apply maxpooling. In this model, we use a total of 3 convotional layers.

*Step 3.* Before fully connect among layers, we flat them with classifier.add(flatten())

*Step 4.* Finally, we put all the layers together into a completely CNN [5]. 2 final layer is a fully connected layer. This layer connect every neuron from max-pooled layer to every output neuron. At this point, we have a CNN [5] used for training.



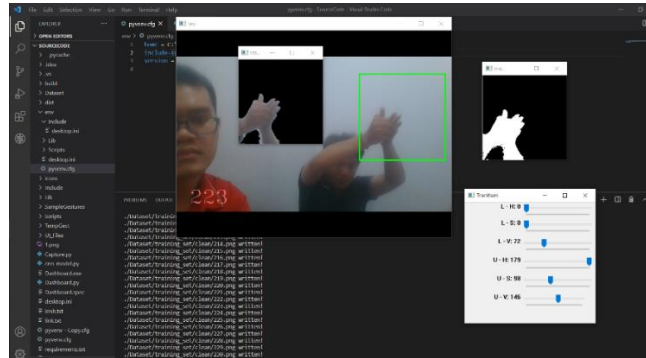
**Fig. 6.** Deep learning architecture of CNN in the context of transportation.

**After creating a CNN [5], we jump to training part with `compile()` method.**

*Step 1.* We specify the training configuration (optimizer, loss, metrics) by `classifier.compile`

*Step 2.* We call `fit()`, which will train the model by slicing the data into "batches" of size "batch\_size", and repeatedly iterating over the entire dataset for a given number of "epochs". The Keras library [4] will help retraining our CNN [5]. At this point, we have a completely trained model [5].

All the models we trained are converted by `classifier.save` into h5py file (VSLModel.h5) which is a lot lighter than the number of frames input that we put in, just encapsulated in one model file.

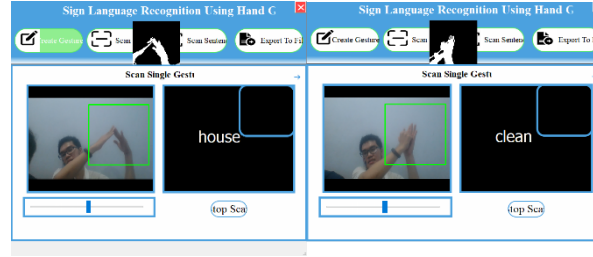


**Fig. 7.** Training by CNN.



#### 4.5 Recognition of Real-time camera.

The problem when we compare real time is that we need to compare data sources, so we use deep learning to do that. As we mentioned the above image processing method, we also use it for handling the input gestures. We use OpenCV for the webcam recognition. The input gestures are converted into grayscale mode.



**Fig. 8.** Single word detected in real-time.

At this point, we exploit the comparison method of Keras (Tensorflow framework) [4] which can connect with trained data. Our real-time input has size of  $64 \times 64$ , we used `img_to_array` to encode it into matrix of 0 and 1 and into vectors then we start to compare with our trained model (VSLModel.h5) by `classifier.predict`. With the accuracy accepted, the result text is appeared in the screen. However, we set up the gap among words based on the delay of our input gestures. For example, if we don't make any operation or we halt our action for a moment, the program understand it as a delay and the output will be an underscore. The resting frame we set here is 20. We also make a create gestures option which can allow to create your own words but only with low accuracy (1 image to compare) in 1 session.

Our last part is to make an application that everyone can use easily so we use PyQt5 to design a Graphic User Interface.

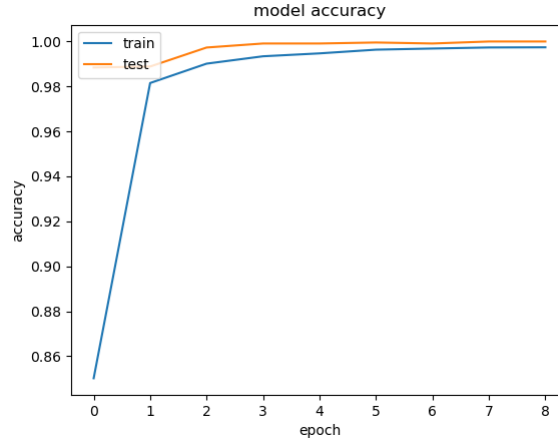
Finally, we execute our project with Pyinstaller. Furthermore, we pack it to 1 file installation which people can use it easily even without installing Python.

## 5 Experimental result

For single gesture tested, we received high accuracy result. Therefore, it is supposed to say that the trained data has worked properly. This is real-time testing, with the latency less than 0.5s. However, the brightness affected a lot in the results since we must convert real-time gestures into grayscale. Furthermore, our project has a limitation which can allow the gestures with the clean background only. If we have a complicated background, for example, our body, the real-time input is disturbed.

For the training accuracy test, we captured 200 frames of a real-time gesture and start to test the accuracy compare with data trained, the accuracy oscillate between 99% and 100%, which can show that our training method has a very high accuracy. Our model is focusing a lot of attention on training data and does not bring generality on

never-before-seen data. This leads to the model achieving extremely good results on the training data set



**Fig. 9.** Training accuracy plot using matplotlib library.

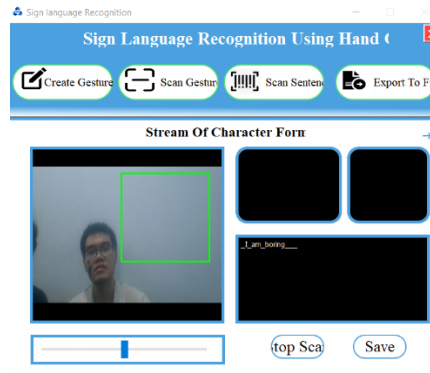
In manual testing, we perform 10 gestures in 200 times and record the obtained results and calculate to percentage accuracy, the accuracy oscillate between 97% and 99.5%. This test is performed on only one camera, the actual accuracy may vary with poor camera quality.

**Table 2.** Result in manual testing

Word	Correct	Accuracy
I	198	99.0%
YOU	198	99.0%
HELP	197	98.5%
PLAY	198	99.0%
CLEAN	197	98.5%
HOUSE	198	99.0%
SOCCER	199	99.5%
BORING	194	97.0%
AM	198	99.0%

For the sentence mode, the project has worked with the accuracy around 95% (statistics 200 tests). We face the problem moving from one word to another, which is the set up of delay recognition. If we do too fast, there is no underscore among words,

otherwise if we do too slow, there are a lot of underscores among words. The resting frames are 20 so its quite sensitive.



**Fig. 10.** “I am boring” recognition

Overall, the application works in real-time smoothly and acceptable. However, we are doing with the low trained data. With the increase of trained data, the accuracy is changed. We will work more to handle all the existed problems. Thus, we will try to go a step further with the addition of background and background filtering.

## 6 Conclusion

After doing research in deep learning and computer vision, we have created a simple, inexpensive application to track hand gestures for translating sign language into plain writing. The program uses conventional cameras to recognize sign language, therefore optimizing costs and simplifying installation and application process. In this project we only sample a certain number of words due to the limitations on our computer hardware used for deep learning. This also affects usability as the trained data is not yet extensive. Of course the most obvious application of this program is that it helps a lot for sign language translation. This program can also be developed to be applied to pre-recorded videos to make content from disabled people more accessible. The program can be applied in communicating with users of sign language in a more natural way if integrated into portable devices such as smart phones, smart glasses. If used in conjunction with video communication programs, it will be more convenient. The project can expand the sign language of countries around the world, thereby helping the mute and deaf have more opportunities to interact with foreigners. Without limiting sign language recognition, programs can be developed to recognize other hand gestures, thus being applied in many other areas such as communicating with a computer through gestures, Artificial Intelligence etc. The app works effectively only under certain environmental lighting conditions and this looks rude and primitive at first. However, we strongly believe that in terms of future prospects.

In our system, we are developing a module which can help to make a sound of the sentences. Although this sound system have not complete yet, it can generate sound of the simple sentence. We hope this application can support for disable people interacting with everyone.

## References

1. Gowri.D, Vidhubal.D (2014). "Sign language recognition for deaf and dumb people". Int J Res Eng Technol, 03(19), pp. 797–799.
2. Neel Kamal Bhagat: Indian Sign Language Gesture Recognition using Image Processing and Deep Learning. 2019 Digital Image Computing: Techniques and Applications (DICTA), Australia (2020).
3. Mr. Shadab Shaikh.: Sign Language Recognition Using Hand Gestures Keras PyQT5 OpenCV. Department of Computer Engineering Anjuman-I-Islam's Kalsekar Technical Campus Affiliated to Mumbai University.
4. "Keras", <https://keras.io/about/>, last accessed 2020/07/30.
5. CS231N Staff, Stanford University, "Convolutional Neural Networks for Visual Recognition", <https://cs231n.github.io/convolutional-networks/>, last accessed 2020/08/08.
6. "Deep Learning - MATLAB & Simulink", <https://www.mathworks.com/solutions/deep-learning/>, last accessed 2021/05/30.
7. Xiaolei Ma, Zhuang Dai , Zhengbing He , Jihui Ma, Yong Wang and Yunpeng Wang: Learning Traffic as Images: A Deep Convolutional Neural Network for Large-Scale Transportation Network Speed Prediction. School of Transportation Science and Engineering, Beijing Key Laboratory for Cooperative Vehicle Infrastructure System and Safety Control, Beihang University, Beijing 100191, China.
8. Suharjito MT, "Sign Language Recognition Application Systems for Deaf-Mute People A Review Based on Input-Process-Output", 2017 [Online]. Available: [https://www.academia.edu/35314119/Sign\\_Language\\_Recognition\\_Application\\_Systems\\_for\\_Deaf-Mute\\_People\\_A\\_Review\\_Based\\_on\\_Input-Process-Output](https://www.academia.edu/35314119/Sign_Language_Recognition_Application_Systems_for_Deaf-Mute_People_A_Review_Based_on_Input-Process-Output) [Accessed April 06, 2019].
9. NAD, "American sign language-community and culture frequently asked questions", 2017 [Online]. Available: <https://www.nad.org/resources/american-sign-language/community-and-culture-frequently-asked-questions/> [Accessed April 06, 2019].
10. NIDCD, "american sign language", 2017 [Online]. Available: <https://www.nidcd.nih.gov/health/american-sign-language>, [Accessed April 06, 2019].