# MINOR-1 PROJECT

**FINAL REPORT**

For

Project Title – Sign Language Recognition System

Submitted By

|  |  |  |
| --- | --- | --- |
| **Specialization** | **SAP ID** | **Name** |
| Artificial Intelligence and Machine Learning | 500082942 | Raghav Akshat Upadhyay |
| Cloud Computing and Virtualization Technology | 500083018 | Rakshit Pratap Singh |



Department of Informatics

School Of Computer Science

UNIVERSITY OF PETROLEUM & ENERGY STUDIES,

DEHRADUN- 248007. Uttarakhand

Dr Surbhi Saraswat Dr T.P Singh

**Project Guide Cluster Head**

**Final Report**

**Project Title –** Sign Language Recognition System

**Abstract**

The act of communicating or exchanging information, ideas, or feelings is referred to as communication. Both parties must be able to speak and understand the same language in order for two people to create communication. Deaf and dumb persons, however, use various communication methods. Inability to hear or speak is referred to as being deaf or dumb. They use sign language to communicate with each other and with everyday people, yet everyday people do not value sign language as much as they should. Communication between a normal person and a deaf or dumb person is challenging since not everyone is familiar with or understands sign language.

The two emerging topics of study are pattern recognition and gesture recognition. Hand gestures play a crucial role in nonverbal communication and are essential to daily living. With the aid of a hand gesture detection system, we have access to a novel, comfortable, and user-friendly method of interacting with computers that is more suited to human needs.

**Keywords:** American Sign Language (ASL), Sign Language Recognition; Convolutional Neural Network (CNN).

**1. Introduction**

Deafness is a handicap that impairs hearing and renders a person mute, and muteness is a disability that impairs speech and renders a person mute. Both can still do a lot of other things because their hearing and/or speaking is the only thing that is impaired. Communication is the only thing separating them from average people. The deaf-mute people can live easily as a regular person if there is a mechanism for normal people and deaf-mute people to communicate. And sign language is the only means of communication for them.

Despite the fact that sign language is crucial for deaf-mute persons to communicate with others and with themselves, normal people nevertheless pay it little mind. Unless we have relatives who are deaf-mute, we as normal people often overlook the value of sign language. Using a sign language interpreter’s services is one way to communicate with deaf-mute people. However, hiring a sign language interpreter might be expensive. For the deaf-mute and normal individuals to converse normally, a low-cost solution is required.

Therefore, scientists are working to develop a means of communication for deaf-mute persons so that they can interact with hearing people. The Sign Language Recognition System is the innovation in this. The technology seeks to understand sign language and translate it either orally or in text form into the native tongue. But there are a number of issues with establishing sign language recognition, from picture acquisition to categorization. The ideal technique for acquiring images is still being investigated by scientists. The challenges of picture pre-processing are presented by image collection using a camera. Using an active sensor device, meanwhile, can be expensive. Researchers' use of classification systems has certain downsides as well. Researchers are unable to choose the optimum recognition method because there are so many options. By concentrating on one method, it tends to prevent testing of other methods that might be more appropriate for sign language recognition. We aim to use Convolutional Neural Networks to perform our classification.

**2. Literature Review**

Sign languages are defined as an organized collection of hand gestures having specific meanings which are employed from the hearing impaired people to communicate in everyday life [3]. Being visual languages, they use the movements of hands, face, and body as communication mediums. There are over 300 different sign languages available all around the world [5]. Though there are so many different sign languages, the percentage of population knowing any of them is low which makes it difficult for the specially-abled people to communicate freely with everyone. SLR provides a means to communicate in sign language without knowing it. It recognizes a gesture and translates it into a commonly spoken language like English. SLR is a very vast topic for research where a lot of work has been done but still various things need to be addressed. The machine learning techniques allow the electronic systems to take decisions based on experience i.e. data. The classification algorithms need two datasets – training dataset and testing dataset. The training set provides experiences to the classifier and the model is tested using the testing set [6]. Many authors have developed efficient data acquisition and classification methods [3][7]. Based on data acquisition method, previous work can be categorized into two approaches: the direct measurement methods and the vision-based approaches [3]. The direct measurement methods are based on motion data gloves, motion capturing systems, or sensors. The motion data extracted can supply accurate tracking of fingers, hands, and other body parts which leads to robust SLR methodologies development. The vision-based SLR approaches rely on the extraction of discriminative spatial and temporal from RGB images. Most of the vision-based methods initially try to track and extract the hand regions before their classification to gestures [3]. Hand detection is achieved by semantic segmentation and skin colour detection as the skin colour is usually distinguishable easily [8][9]. Though, because the other body parts like face and arms can be mistakenly recognized as hands, so, the recent hand detection methods also use the face detection and subtraction, and background subtraction to recognize only the moving parts in a scene [10][11]. To attain accurate and robust hands tracking, particularly in cases of obstructions, authors employed filtering techniques, for example, Kalman and particle filters [10][12].

**3. Problem Statement**

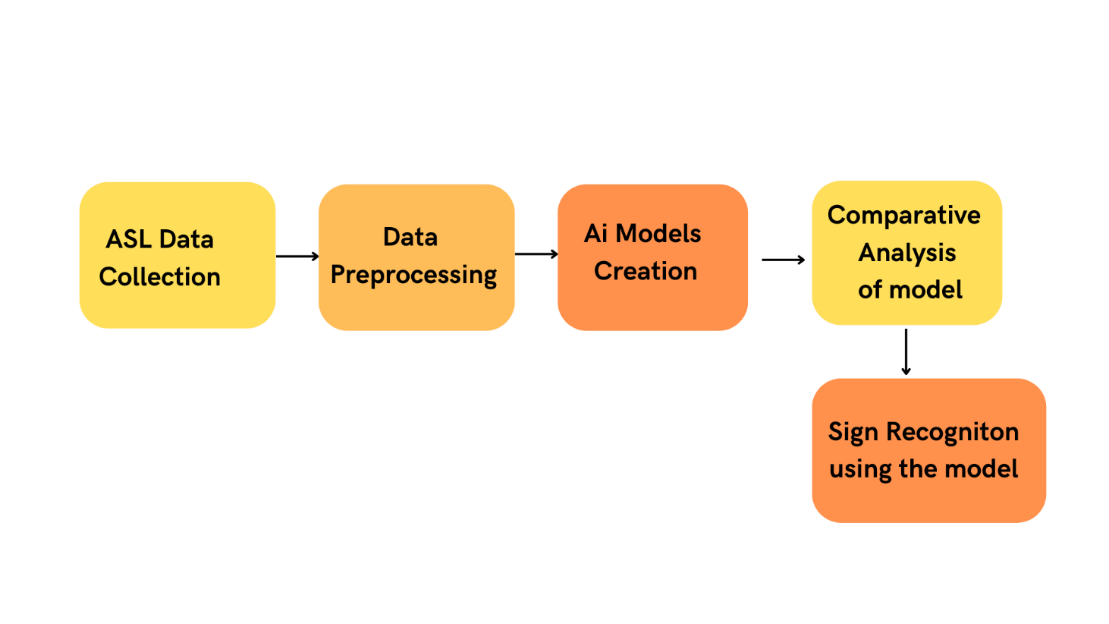
Completing our literature review, we discovered that isolated Sign Language Recognition are the best ways to go about even if they are hard to train they let the user and the consumer access to the entire alphabets. It has been a popular and difficult area of research to identify sign language motions in real-time video and correctly classify them into one of a number of categories. This field has been the focus of numerous researchers for a long time. This idea has also been the subject of research by Liang et al. [6], which has helped us throughout the implementation. The single-sentence characterization of the work carried out by this suggested system is the process of identifying and categorising a sign language gesture. Additionally, there is a text to ASL finger spelling capability that enables two-way communication from signing to text. Python and OpenCV are used to capture webcam photos for data acquisition. Real-time computer vision is the primary focus of OpenCV's functions. It expedites the incorporation of artificial intelligence into commercial goods and offers a standard infrastructure for computer vision-based applications. The alphabet signs that make up the dataset were developed to create the dataset, 25 photos for each letter of the alphabet are taken. A buffer of five seconds is provided between two specific signs, i.e., to go from the sign of one alphabet to the sign of a different alphabet, and photographs are taken every two seconds, giving time to record gesture with a little bit of variation every time.

**4. Objectives**

The main objective of our project is to develop a Sign Language Recognition system which will take in images from the users a camera and use it to classify using CNN and fine tune it and classify them into one of the 24 labels. Python and OpenCV are used to capture webcam photos for data acquisition. Real-time computer vision is the primary focus of OpenCV's functions. It expedites the incorporation of artificial intelligence into commercial goods and offers a standard infrastructure for computer vision-based applications. The alphabet signs that make up the dataset were developed To create the dataset, 24 photos for each letter of the alphabet are taken. A buffer of five seconds is provided between two specific signs, i.e., to go from the sign of one alphabet to the sign of a different alphabet, and photographs are taken every two seconds, giving time to record gesture with a little bit of variation every time.

**5. Methodology**

The overall design algorithm of the project consists of five components:



The figure describes the overall steps involved in this project.

Fig.1 Project model workflow

We plan to create a real-time sign language detector utilising the TensorFlow and Py-Torch object detection APIs and train it using transfer learning on the provided dataset. Python and OpenCV are used to capture webcam photos for data acquisition. Real-time computer vision is the primary focus of OpenCV's functions. It expedites the incorporation of artificial intelligence into commercial goods and offers a standard infrastructure for computer vision-based applications. The alphabet signs that make up the dataset were developed To create the dataset, 25 photos for each letter of the alphabet are taken. A buffer of five seconds is provided between two specific signs, i.e., to go from the sign of one alphabet to the sign of a different alphabet, and photographs are taken every two seconds, giving time to record gesture with a little bit of variation every time. Following the data collection, a labelled map is produced that represents all the model's objects and includes their id and the label for each sign in the alphabet. Each of the 24 labels on the label map corresponds to a letter of the alphabet. Each label has a distinct ID that ranges from 1 to 24. To seek up the class name, use this as a reference.

Dataset: The original MNIST image dataset of handwritten digits is a popular benchmark for image-based machine learning methods but researchers have renewed efforts to update it and develop drop-in replacements that are more challenging for computer vision and original for real-world applications.

The American Sign Language letter database of hand gestures represent a multi-class problem with 24 classes of letters (excluding J and Z which require motion). The dataset format is patterned to match closely with the classic MNIST. Each training and test case represents a label (0-25) as a one-to-one map for each alphabetic letter A-Z (and no cases for 9=J or 25=Z because of gesture motions). The training data (27,455 cases) and test data (7172 cases) are approximately half the size of the standard MNIST but otherwise similar with a header row of label, pixel1, pixel2 to pixel784 which represent a single 28x28 pixel image with grayscale values between 0-255. The original hand gesture image data represented multiple users repeating the gesture against different backgrounds. The Sign Language MNIST data came from greatly extending the small number (1704) of the color images included as not cropped around the hand region of interest. We can check a random sample of the training data in Fig 2

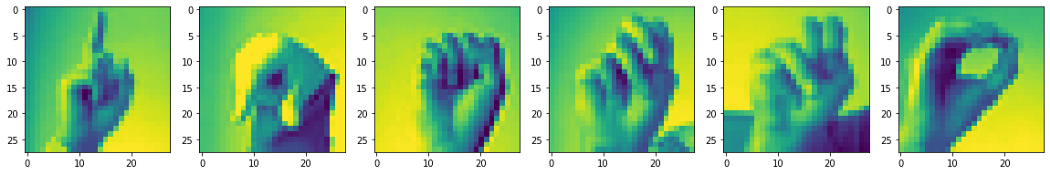


Fig 2. Random sample from dataset



Fig 1: American Sign Language label with pictures

In order to train a convolutional neural network with the ASL alphabet dataset, we first create a train and test dataset containing labels for both in NumPy array format. A class of deep, feed-forward artificial neural networks called convolutional neural networks are most frequently used to evaluate visual imagery. A multilayer perceptron variant used by CNNs is made to require less pre-processing. Which we will try to fine tune using augmentation, batch-normalization and dropout layer a Neural Network Model [7] has established itself as a significant advancement in the field of image detection and classification.

**6. PERT Chart**

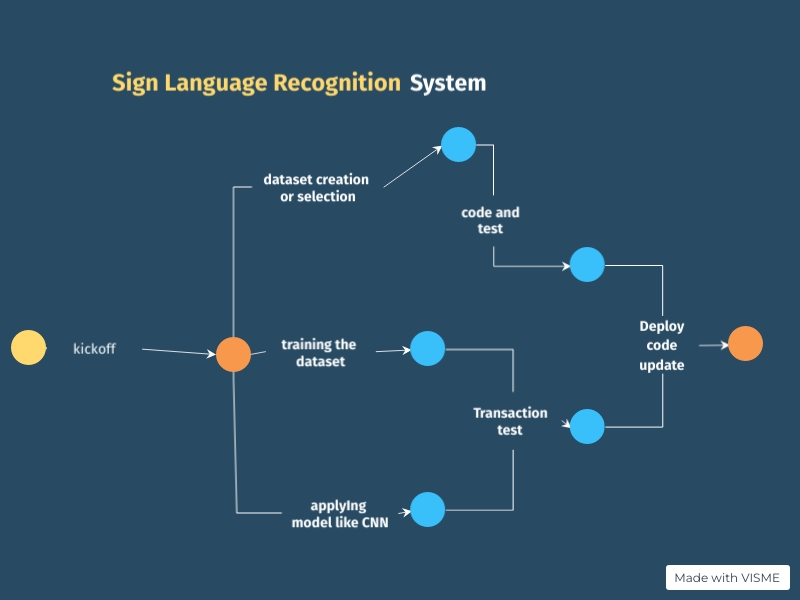
****

Fig 3. PERT Chart

**Timeline**

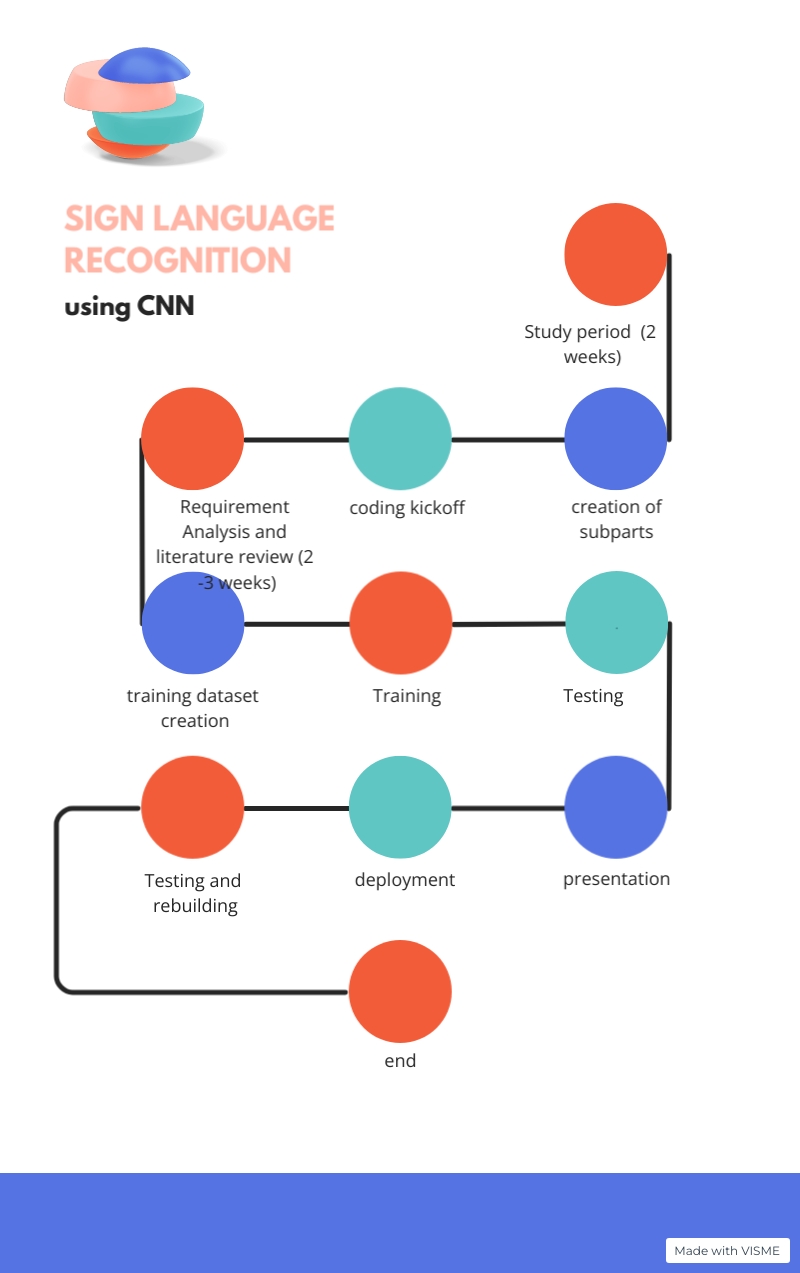
****

Fig 4. Timeline

**7.Progress:**

**Data Pre-processing:**

Preprocessing data is a common first step in any project to prepare raw data in a format that the AI model can accept. For example, we’ll be converting the image data to pixel format. we preprocess data to enhance desired features or reduce artifacts that can bias the AI model.

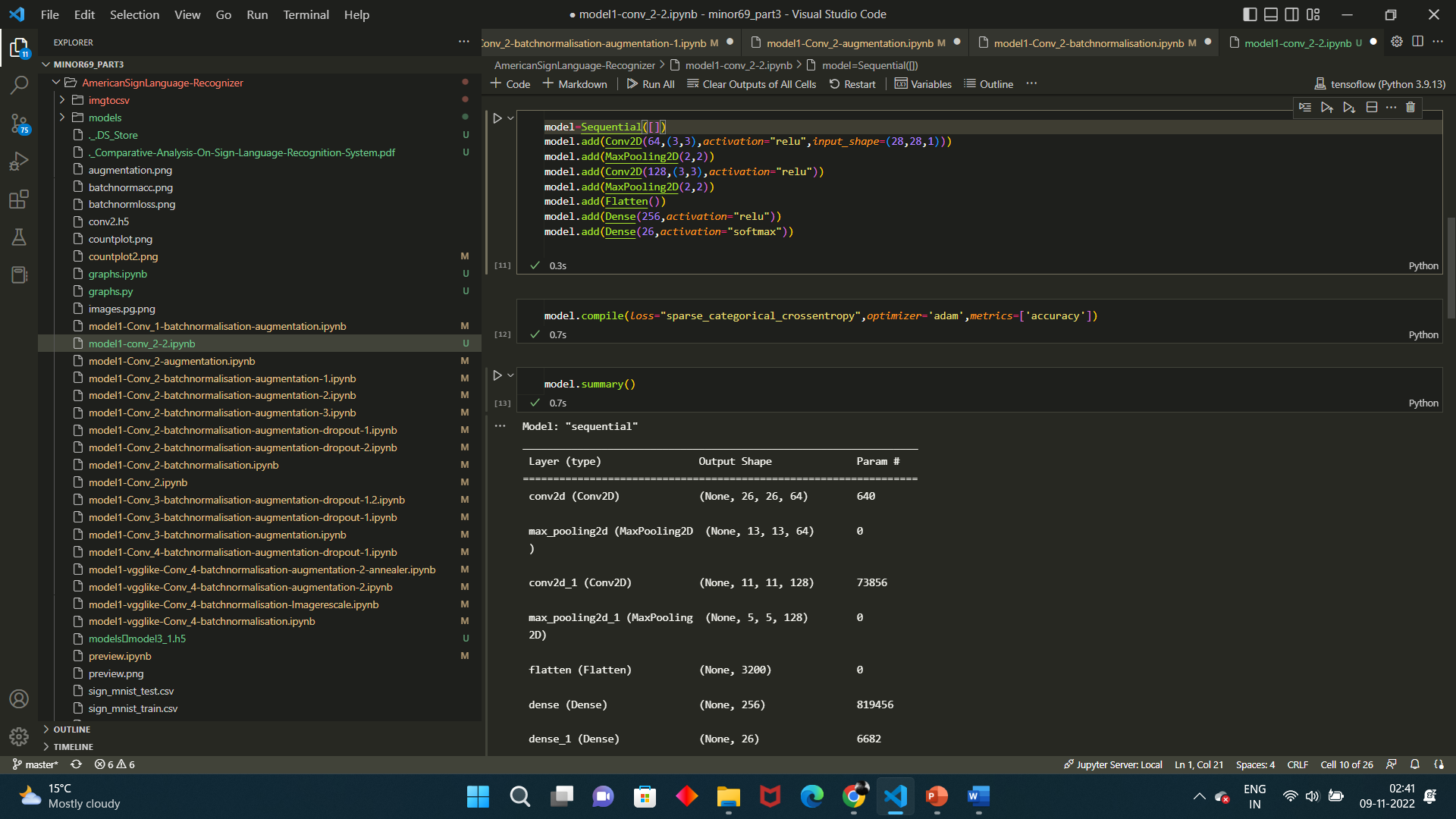
We have done the processing in our data as the input layer of the model will take images of size (28,28,1) where 28,28 are height and width of the image respectively while 1 represents the colour channel of the image for grayscale.

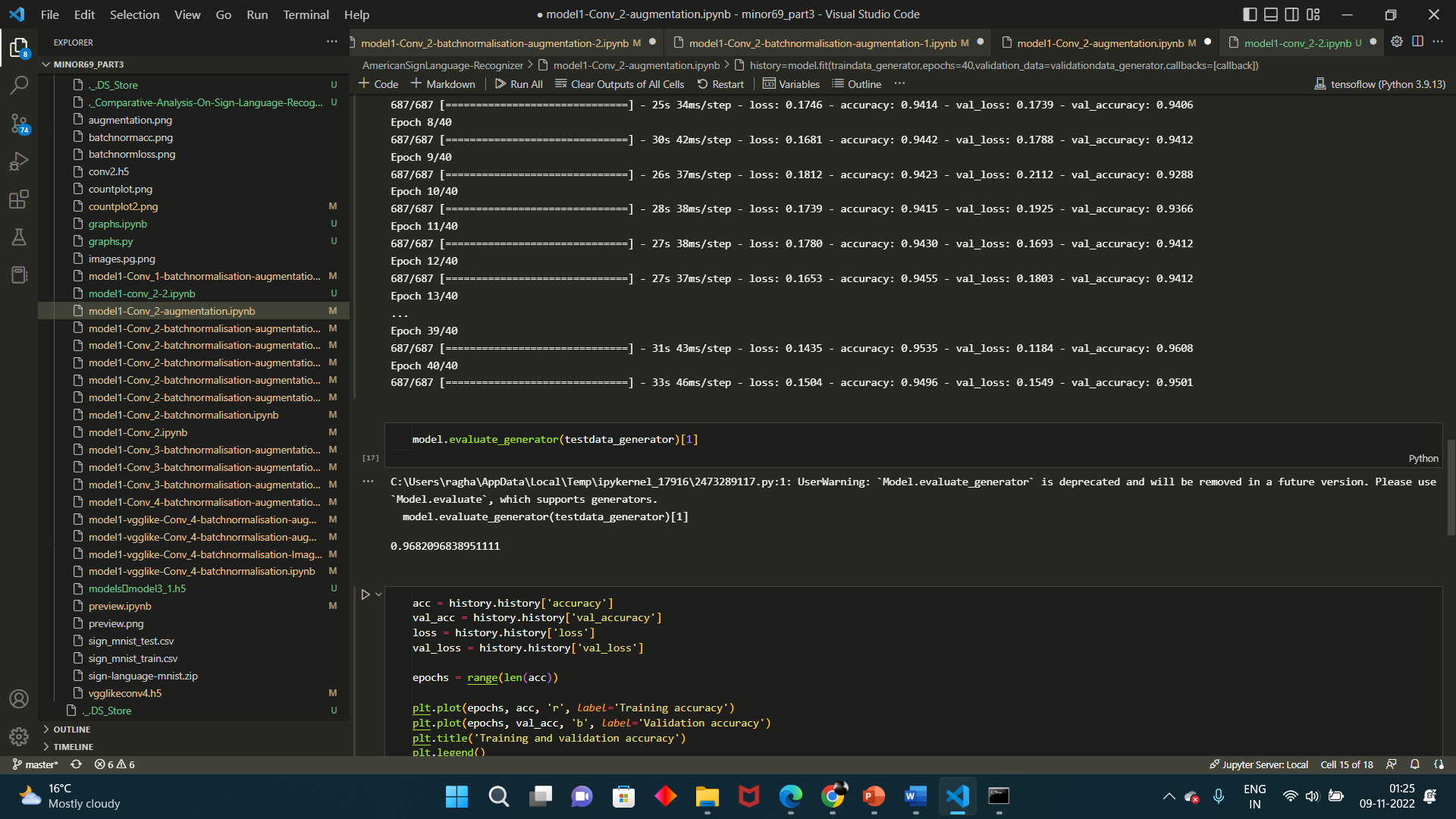
We can do it either with the help reshape function of the tensor-flow library later we can add augmentation with the Image-data-generator class.

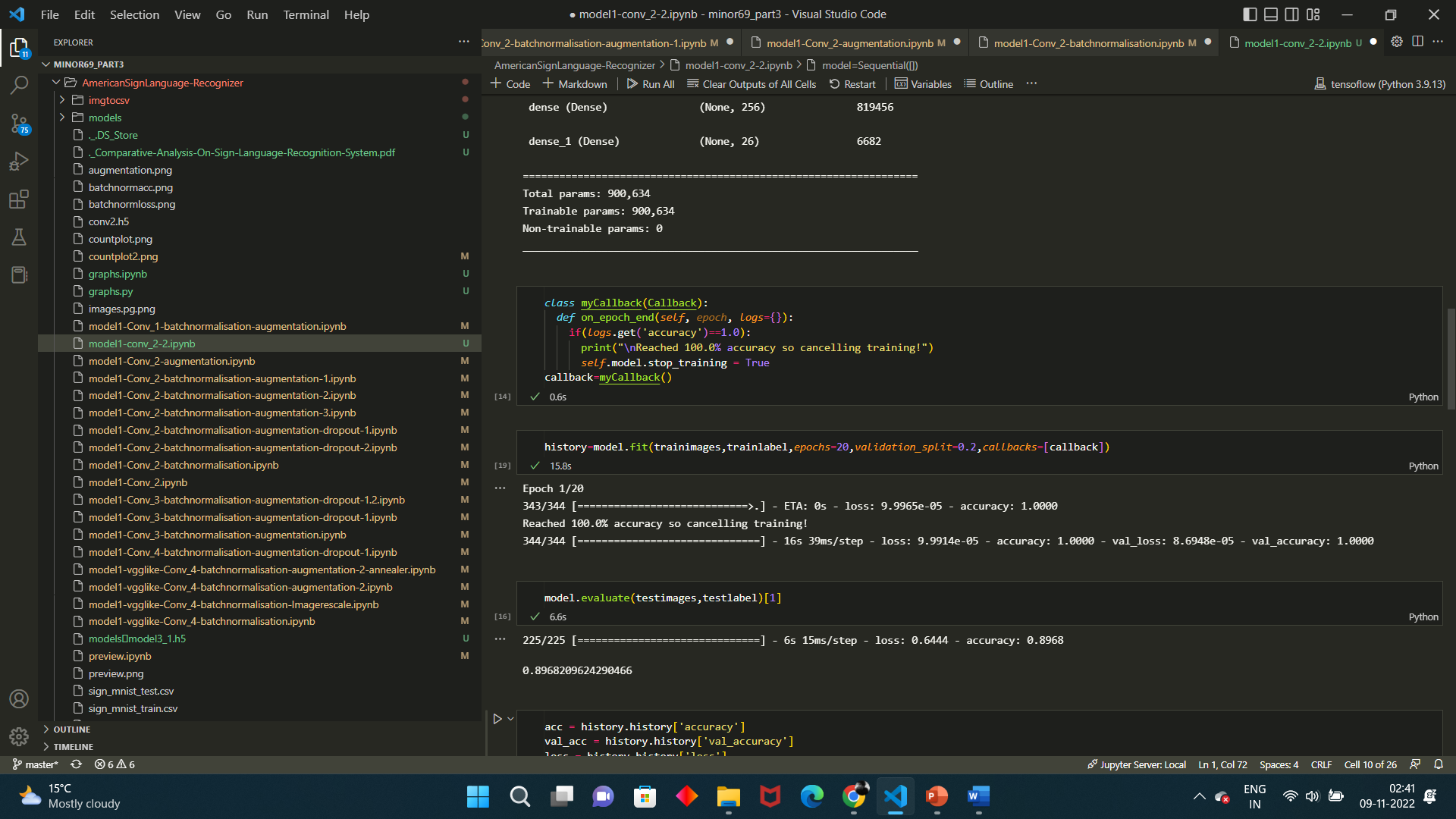
**Model Creation:**

The output layer of the model will have 26 neurons for 26 different letters, and the activation function will be soft-max since it is a multiclass classification problem.

Intially we will use 2 convolutional layers



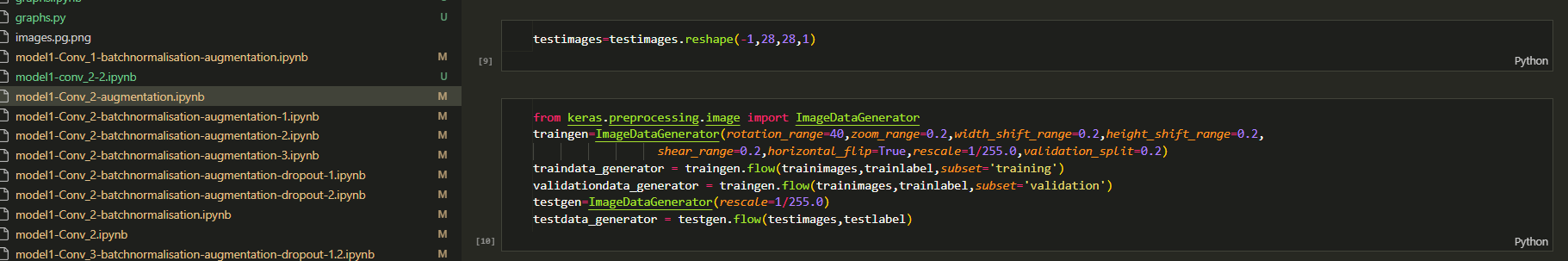




The Training Accuracy for the Model is 100% while test accuracy for the model is 91%. This is clearly an overfitting situation.

In the next step, we use Data Augmentation to solve the problem of overfitting.

Data Augmentation: There can be some features/orientation of images present in the test dataset that are not available in the training dataset. And Hence, our model is unable to identify those patterns. This is can be solved by augmenting the data. Data Augmentation is an essential step in training the neural network. For example, in the training dataset, we have hand signs of the right hands but in the real world, we could get images from both right hands as well as left hands. Data Augmentation allows us to create unforeseen data through Rotation, Flipping, Zooming, Cropping, Normalizing etc. Tensor-flow provides an Image-Data-Generator function which augments data in memory on the flow without the need of modifying local data. This also gives us the room to try different augmentation parameters. We will Augment the data and split it into 80% training and 20% validation.

****

****

After Augmenting the data, the training accuracy after 40 epochs is 94.5% and test accuracy is at around 94.8 %. And after 100 epochs the accuracy is 93.5 for training and 97.8 for testing



but 100 epochs are a lot of epochs so we need good results in less number or epochs so we can use batch-normalization.

Batch Normalisation: Batch Normalisation allows normalising the inputs of the hidden layer. As from the above model, we can see that though, with data augmentation, we can resolve overfitting to training data but requires more time for training. Batch Normalisation resolves this issue, by normalising the weights of the hidden layer.

The training accuracy after batch normalization is about 95.50 % for training and 94.50 for testing and it also requires way less no of epochs

**Accuracies for different architecture of CNN**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Model | No. of Conv layers | Augmentation | Batch Normalisation | Dropout | Training Accuracy | Test Accuracy |
| 1 | 1 | yes | yes | yes | 97.2 | 96.12 |
| 2 | 2 | no | no | no | 96 | 91.2 |
| 3 | 2 | yes | no | no | 94.96 | 96.82 |
| 4 | 2 | no | yes | no | 100 | 94.72 |
| 5 | 2 | yes | yes | no | 94.4 | 94.6 |
| 6 | 2 | yes | yes | no | 98 | 98.2 |
| 7 | 2 | yes | yes | no | 98 | 95.83 |
| 8 | 2 | yes | yes | yes(0.4) | 94.008 | 96 |
| 9 | 2 | yes | yes | yes(0.4,0.4) | 84.5 | 97.54 |
| 10 | 3 | yes | yes | no | 98.06 | 98.3 |
| 11 | 3 | yes | yes | 0.4 | 90 | 93 |
| 12 | 3 | yes | yes | 0.2 | 92.85 | 97.57 |
| 13 | 4 | no | yes | no | 98 | 96.29 |
| 14 | 4 | yes | yes | no | 98.06 | 98 |
| 15 | 4 | yes | yes | 0.4 | 98.9 | 97.4 |

**Real World Validation Set Testing**

Following this we created a real time generator class to take images and test them in real time to do so we created a small data set of 200 images on all alphabets so we could test them using the architecture that gives the best accuracy.

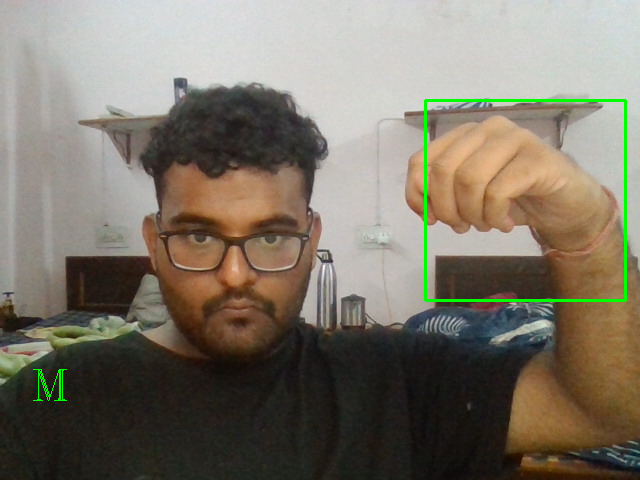


Fig 3.Real time testing for the letter M

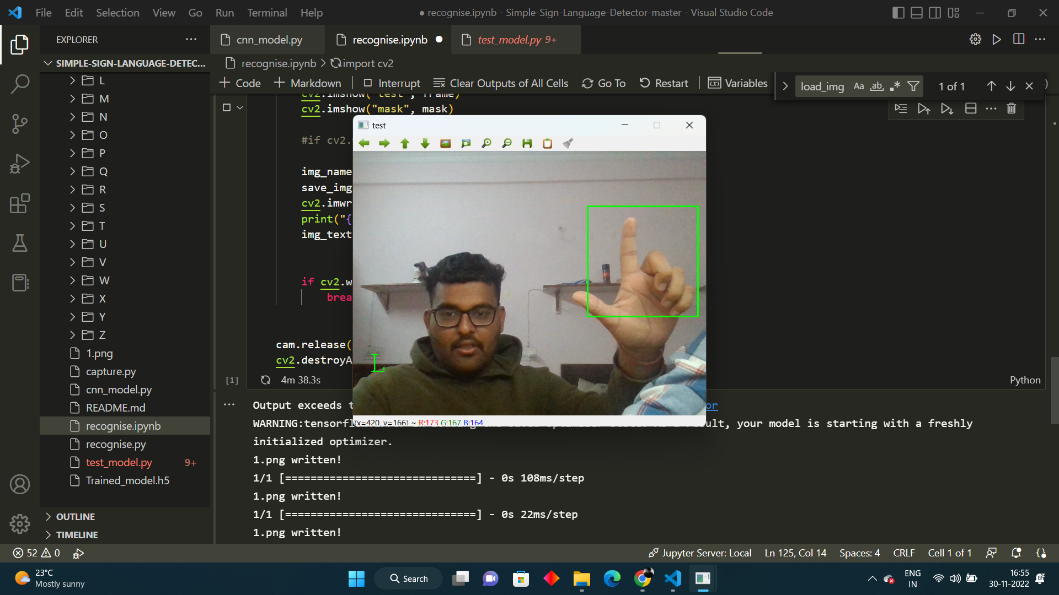


Fig 4.Real time testing for the letter L

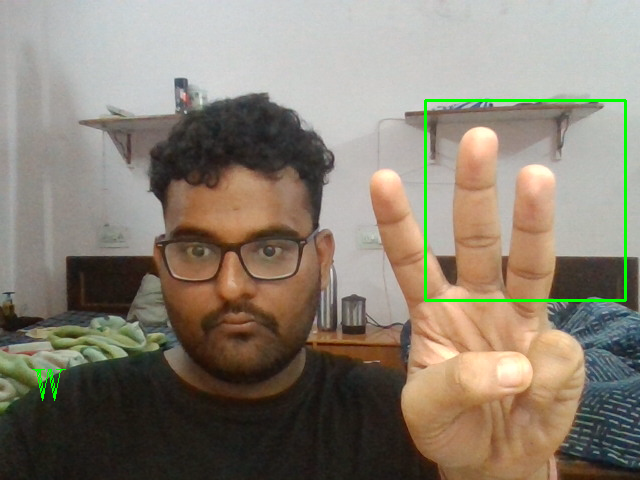


Fig 4.Real time testing for the letter W

**8. References**

1. Kapur, R.: The Types of Communication. MIJ. 6, (2020).
2. Suharjito, Anderson, R., Wiryana, F., Ariesta, M.C., Kusuma, G.P.: Sign Language Recognition Application Systems for Deaf-Mute People: A Review Based on InputProcess-Output. Procedia Comput. Sci. 116, 441–448 (2017). <https://doi.org/10.1016/J.PROCS.2017.10.028>.
3. Konstantinidis, D., Dimitropoulos, K., Daras, P.: Sign language recognition based on hand and body skeletal data. 3DTV-Conference. 2018-June, (2018). <https://doi.org/10.1109/3DTV.2018.8478467>.
4. Dutta, K.K., Bellary, S.A.S.: Machine Learning Techniques for Indian Sign Language Recognition. Int. Conf. Curr. Trends Comput. Electr. Electron. Commun. CTCEEC 2017. 333–336 (2018). <https://doi.org/10.1109/CTCEEC.2017.8454988>.
5. Bragg, D., Koller, O., Bellard, M., Berke, L., Boudreault, P., Braffort, A., Caselli, N., Huenerfauth, M., Kacorri, H., Verhoef, T., Vogler, C., Morris, M.R.: Sign Language Recognition, Generation, and Translation: An Interdisciplinary Perspective. 21st Int. ACM SIGACCESS Conf. Comput. Access. (2019). <https://doi.org/10.1145/3308561>.
6. Rosero-Montalvo, P.D., Godoy-Trujillo, P., Flores-Bosmediano, E., Carrascal-Garcia, J., 12 Otero-Potosi, S., Benitez-Pereira, H., Peluffo-Ordonez, D.H.: Sign Language Recognition Based on Intelligent Glove Using Machine Learning Techniques. 2018 IEEE 3rd Ecuador Tech. Chapters Meet. ETCM 2018. (2018). <https://doi.org/10.1109/ETCM.2018.8580268>.
7. Zheng, L., Liang, B., Jiang, A.: Recent Advances of Deep Learning for Sign Language Recognition. DICTA 2017 - 2017 Int. Conf. Digit. Image Comput. Tech. Appl. 2017- Decem, 1–7 (2017). <https://doi.org/10.1109/DICTA.2017.8227483>.
8. Rautaray, S.S.: A Real Time Hand Tracking System for Interactive Applications. Int. J. Comput. Appl. 18, 975–8887 (2011).
9. Zhang, Z., Huang, F.: Hand tracking algorithm based on super-pixels feature. Proc. - 2013 Int. Conf. Inf. Sci. Cloud Comput. Companion, ISCC-C 2013. 629–634 (2014). <https://doi.org/10.1109/ISCC-C.2013.77>.
10. Lim, K.M., Tan, A.W.C., Tan, S.C.: A feature covariance matrix with serial particle filter for isolated sign language recognition. Expert Syst. Appl. 54, 208–218 (2016). <https://doi.org/10.1016/J.ESWA.2016.01.047>.
11. Lim, K.M., Tan, A.W.C., Tan, S.C.: Block-based histogram of optical flow for isolated sign language recognition. J. Vis. Commun. Image Represent. 40, 538–545 (2016). <https://doi.org/10.1016/J.JVCIR.2016.07.020>.
12. Gaus, Y.F.A., Wong, F.: Hidden Markov Model - Based gesture recognition with overlapping hand-head/hand-hand estimated using Kalman Filter. Proc. - 3rd Int. Conf. Intell. Syst. Model. Simulation, ISMS 2012. 262–267 (2012). <https://doi.org/10.1109/ISMS.2012.67>.
13. Nikam, A.S., Ambekar, A.G.: Sign language recognition using image based hand gesture recognition techniques. Proc. 2016 Online Int. Conf. Green Eng. Technol. IC-GET 2016. (2017). <https://doi.org/10.1109/GET.2016.7916786>.
14. Mohandes, M., Aliyu, S., Deriche, M.: Arabic sign language recognition using the leap motion controller. IEEE Int. Symp Ind. Electron. 960–965 (2014). https://doi.org/10.1109/ISIE.2014.6864742.