**Comparative Analysis for Sign Language Recognition using CNN**

Raghav Akshat Upadhyay, Rakshit Pratap Singh

Department of Informatics School of Computer Science

UNIVERSITY OF PETROLEUM & ENERGY STUDIES,

DEHRADUN- 248007. Uttarakhand

**Abstract**

Communication is defined as the act of sharing or exchanging information, ideas or feelings. To establish communication between two people, both of them are required to have knowledge and understanding of a common language. But in the case of deaf and dumb people, the means of communication are different. Deaf is the inability to hear and dumb is the inability to speak. They communicate using sign language among themselves and with normal people but normal people do not take seriously the importance of sign language. Not everyone possesses the knowledge and understanding of sign language which makes communication difficult between a normal person and a deaf and dumb person. To overcome this barrier, one can build a model based on machine learning. A model can be trained to recognize different gestures of sign language and translate them into English. This will help a lot of people in communicating and conversing with deaf and dumb people. The existing American Sign Language Recognition systems are designed using machine learning algorithms with single and double-handed gestures but they are not real-time. In this paper, we propose a method to create an Indian Sign Language dataset using a webcam and then using transfer learning, train a TensorFlow model to create a real-time Sign Language Recognition system using CNN. The system achieves a good level of accuracy even with a limited size dataset.

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

**1. Introduction**

Communication can be defined as the act of transferring information from one place, person, or group to another. It consists of three components: the speaker, the message that is being communicated, and the listener. It can be considered successful only when whatever message the speaker is trying to convey is received and understood by the listener. It can be divided into different categories as follows [1]: formal and informal communication, oral (face-to-face and distance) and written communication, non-verbal, grapevine, feedback, and visual communication, and the active listening. The formal communication (official communication) is steered through the channels that are pre-determined. The unofficial or grapevine communication is the spontaneous communication between individuals in one’s profession that does not have any formal protocol or structure. The oral communication (face-to-face and distance) is the communication in which words are exchanged between people who are present in front or at a distance (with the help of technology including voice and video calls, webinars, etc.). The written communication is the communication in which letters, emails, notices, or any other written form is used for communicating. The non-verbal communication is the communication that uses gestures, facial expressions, body language, etc. The feedback communication happens when a person gives feedback on some product or service provided by an individual or a company. The visual communication occurs when a person gets information from a visual source like televisions, social networking, or any other source. Active listening is when a person listens to and understands what the other individual is trying to convey so that the communication becomes more meaningful and effective [1]. Non-verbal communication helps deaf and dumb people to communicate amongst themselves and with others. Deaf is a disability that impairs a person's hearing ability and makes them incapable to hear while dumb is a disability that impairs the speaking ability and makes them incapable to speak. Not being able to speak or listen makes it difficult to establish communication with others. This is where sign languages come into the role, it enables a person to communicate without words. But a problem still exists, not many people possess the knowledge of sign language. Deaf and dumb may be able to communicate amongst themselves using sign languages but it is still difficult for them to communicate with people having normal hearing and vice-versa due to the lack of knowledge of sign languages. This issue can be resolved by the use of a technology-driven solution. By using such a solution, one can easily translate the gestures of sign language into the commonly spoken language, English. A lot of research has been done in this field and there is still a need for further research. For gesture translation, data gloves, motion capturing systems, or sensors have been used [2]. Vision-based SLR systems have also been developed previously [3]. The existing Indian Sign Language Recognition system was developed using machine learning algorithms with MATLAB [4]. Authors have worked on single-handed and doublehanded gestures. They used two algorithms to train their system, K Nearest Neighbours Algorithm and Back Propagation Algorithm. Their system achieved 93-96% accuracy. Though being highly accurate, it is not a real-time SLR system. The objective of this paper is to develop a real-time SLR system using TensorFlow object detection API and train it using a dataset that will be created using a webcam. The rest of this paper after the introduction is organized as follows. Section 2 presents the related work on the SLR system. Section 3 describes the data acquisition and generation. Section 4 focuses on the methodology of the developed system. Section 5 presents the experimental evaluation of the system, and finally, Section 6 concludes the paper with future work.

**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. Dataset and Acquisition**

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

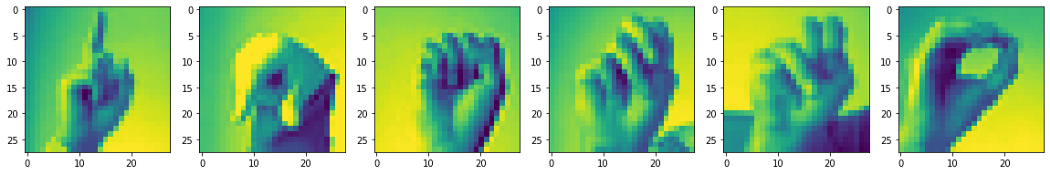


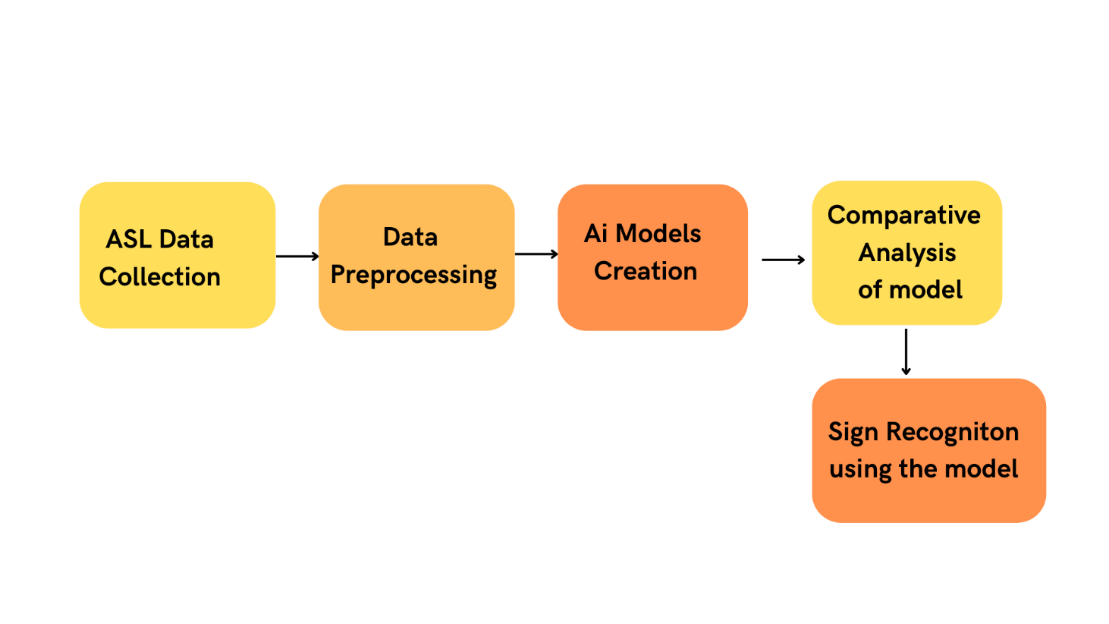
Fig 2. Random sample from dataset

**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.

**4. 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 a 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, 200 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. Following this we will generate models of the following CNN based architecture performing parameter tuning on the dataset to find the best architecture for our model.

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

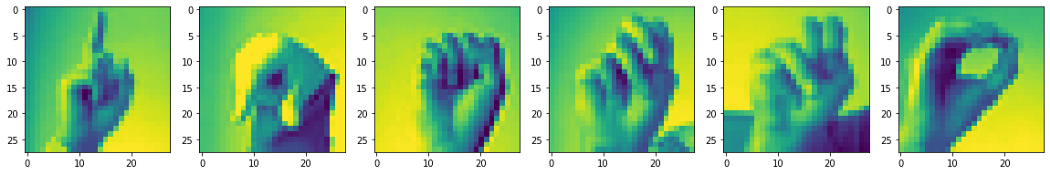


Fig 1. Random sample from dataset



Fig 2: 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.

**5.Results and Discussion**

**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.

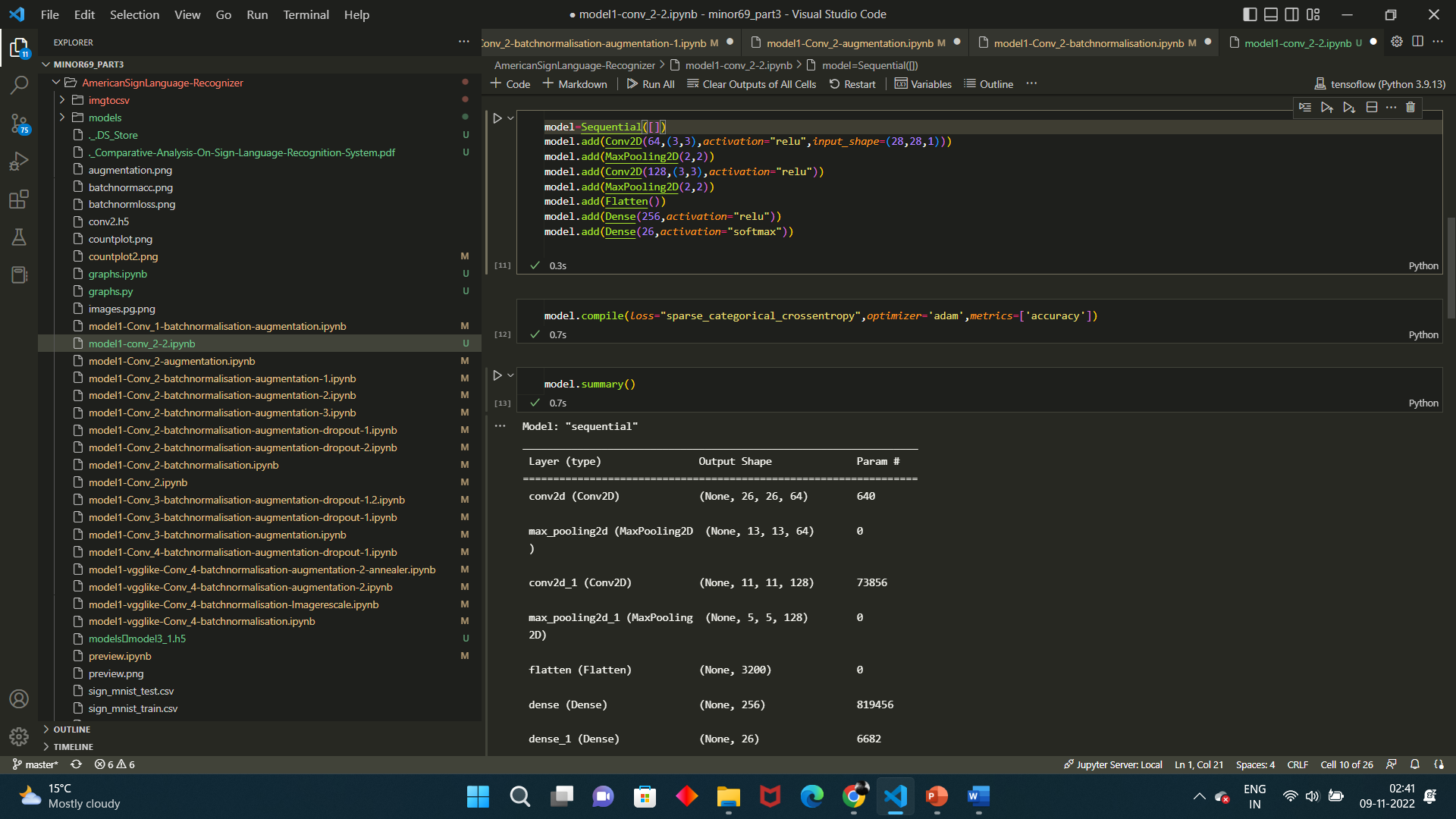
We have done the processing in our data as the input layer of the model will take images of size (28,28,1) where 28,28 are height and width of the image respectively while 1 represents the colour channel of the image for grayscale.

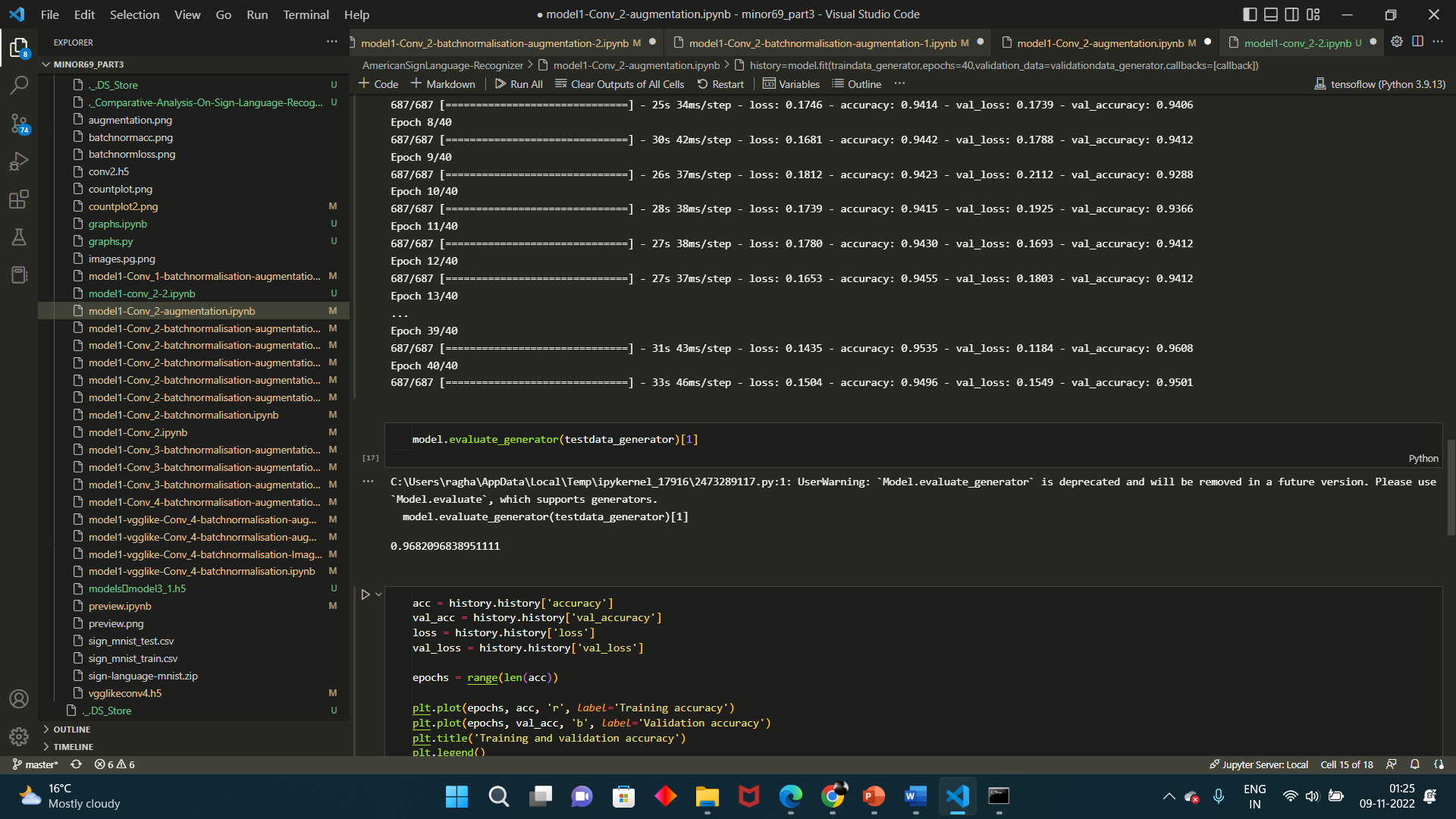
We can do it either with the help reshape function of the Tensorflow library later we can add augmentation with the Image-datagenerator class.

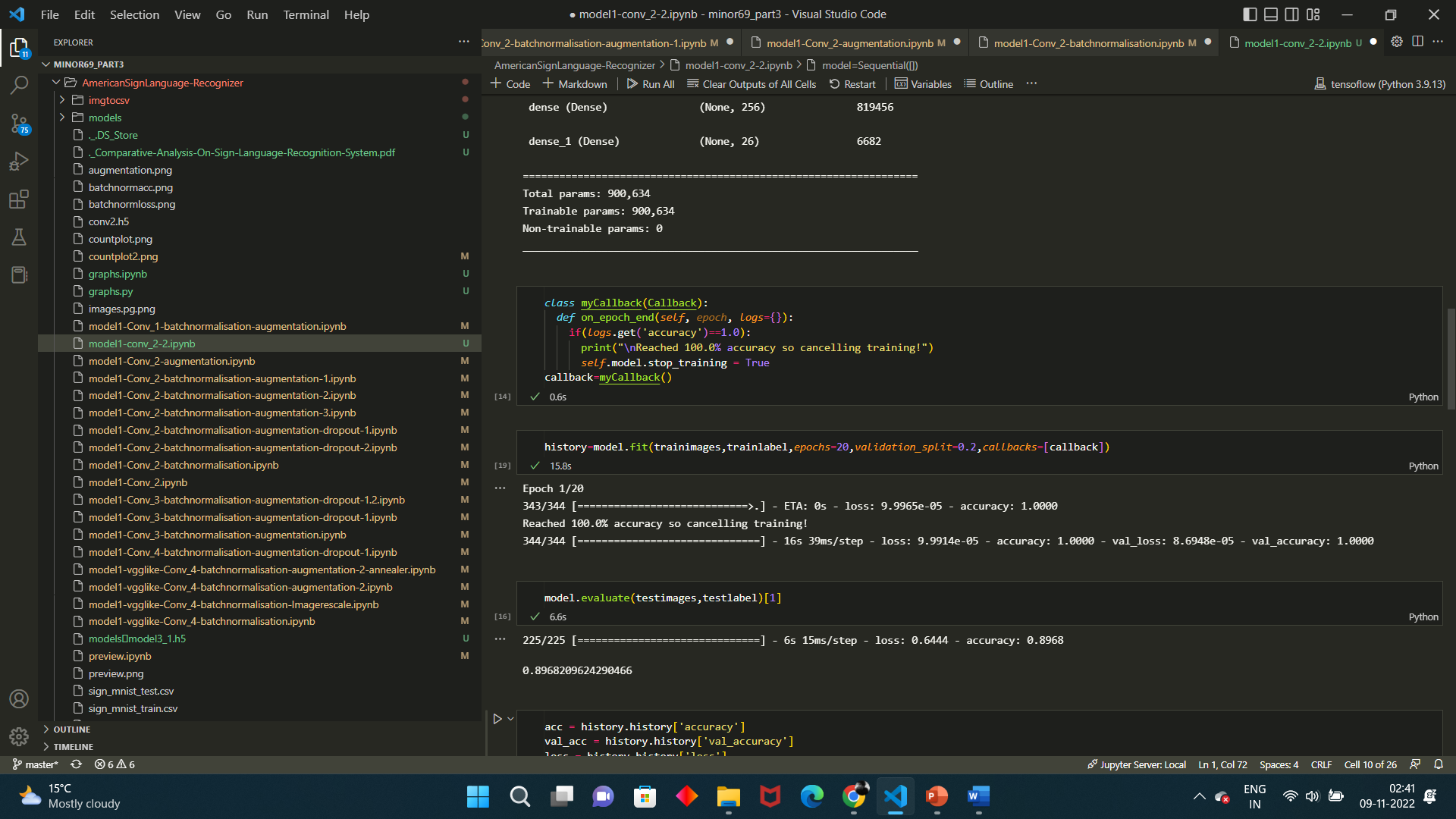
**Model Creation:**

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

Intially we will use 2 convolutional layers



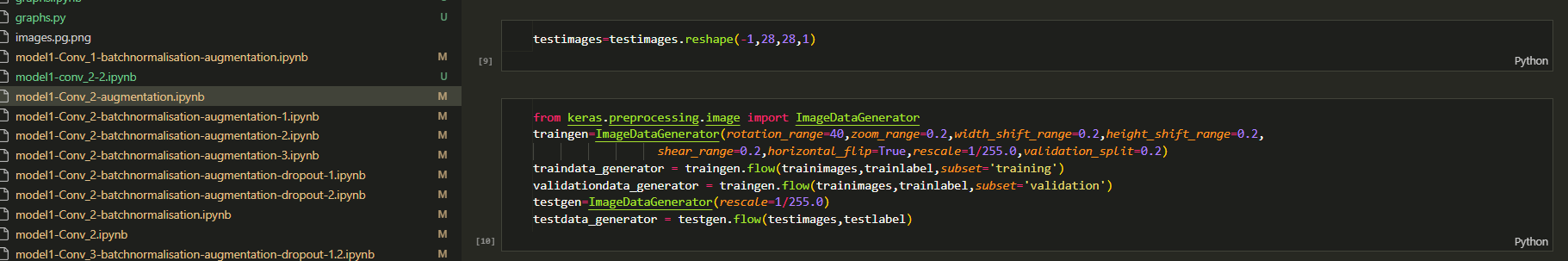




The Training Accuracy for the Model is 100% while test accuracy for the model is 91%. This is clearly an overfitting situation.

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, Normalising etc. Tensorflow provides an ImageDataGenerator 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.

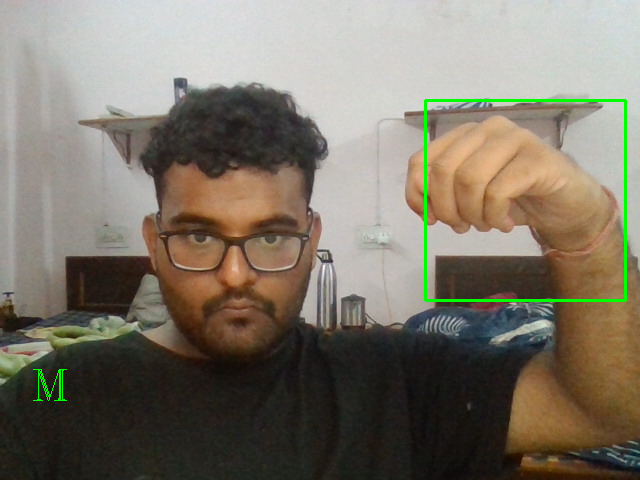


Fig 3.Real time testing for the letter M

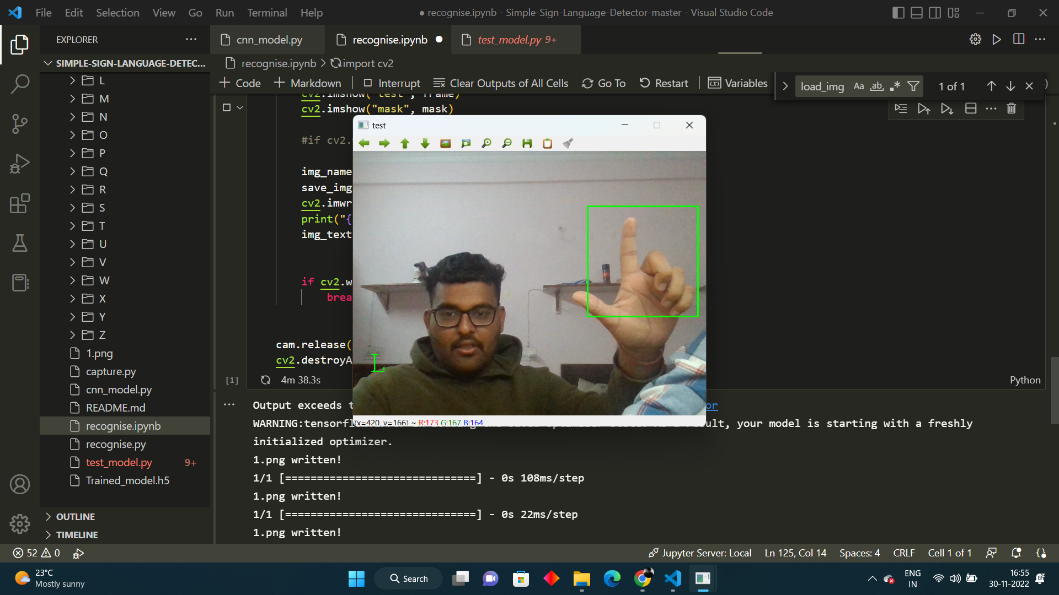


Fig 4.Real time testing for the letter L

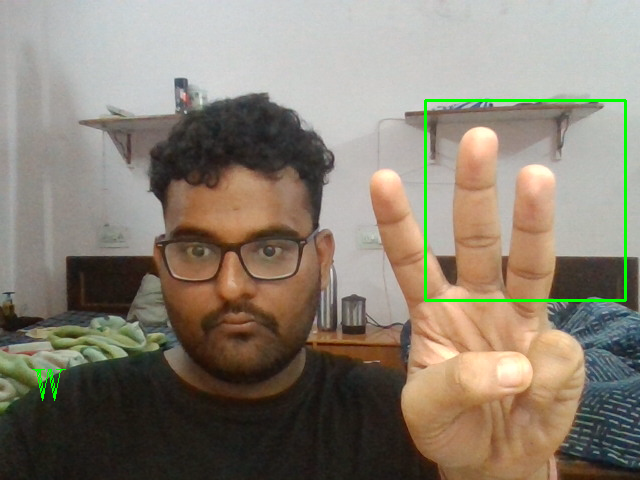


Fig 4.Real time testing for the letter W

**6.Conclusion and Future Works**

Sign languages are kinds of visual languages that employ movements of hands, body and facial expression as a means of communication. Sign languages are important for specially-abled people to have a means of communication. Through it, they can communicate and express and share their feelings with others. The drawback is that not everyone possesses the knowledge of sign languages which limits communication. This limitation can be overcome by the use of automated Sign Language Recognition systems which will be able to easily translate the sign language gestures into commonly spoken language. In this paper, it has been done by TensorFlow object detection API. The system has been trained on the American Sign Language alphabet dataset. The system detects sign language in real-time. For data acquisition, images have been captured by a webcam using Python and OpenCV which makes the cost cheaper. The developed system is showing an average confidence rate of 72.45%. Though the system has achieved a high average confidence rate, the dataset it has been trained on is small in size and limited. In the future, the dataset can be enlarged so that the system can recognize more gestures. The TensorFlow model that has been used can be interchanged with another model as well. The system can be implemented for different sign languages by changing the dataset.

**7.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.