International Journal of Scientific Research in Science and Technology
Available online at : www.ijsrst.com
OPEN ACCESS
Print ISSN: 2395-6011 | Online ISSN: 2395-602X
doi : https://doi.org/10.32628/IJSRST52411223

Sign Language Detection and Recognition Using Media Pipe and Deep Learning Algorithm

[1]Ms. E J Honesty Praiselin, [2]Dr G Manikandan, [3]Ms. Vilma Veronica, [4]Ms. S. Hemalatha
[1]PG Student, Kings Engineering College, Sriperumbudhur, Tamil Nadu, India
[2]Professor, Kings Engineering College, Sriperumbudhur, Tamil Nadu, India, India
[3]Assistant Professor, Kings Engineering College, Sriperumbudhur, Tamil Nadu, India
[4]Assistant Professor, Kings Engineering College, Sriperumbudhur, Tamil Nadu, India

ARTICLEINFO

Article History:
Accepted: 03 March 2024
Published: 13 March 2024

Publication Issue :
Volume 11, Issue 2
March-April-2024
Page Number :
123-130

ABSTRACT

People lacking the sense of hearing and the ability to speak have undeniable communication problems in their life. People with hearing and speech problems communicate using sign language with themselves and others. These communicating signs are made up of the shape of the hand and movement. Sign language is not essentially known to a more significant portion of the human population who uses spoken and written language for communication. Therefore, it is a necessity to develop technological tools for interpretation of sign language. Much research have been carried out to acknowledge sign language using technology for most global languages. But there are still scopes of development of tools and techniques for sign language development for local dialects.

This work attempts to develop a technical approach for recognizing American Sign Language Using machine learning techniques, this work tried to establish a system for identifying the hand gestures from American Sign Language. A combination of two-dimensional and three-dimensional images of Assamese gestures has been used to prepare a dataset. The Media Pipe framework has been implemented to detect landmarks in the images. The results reveal that the method implemented in this work is effective for the recognition of the other alphabets and gestures in Sign Language. This method could also be tried and tested for the recognition of signs and gestures for various other local languages of India

INDEX TERMS Sign language, image recognition, machine learning, features extraction

I. INTRODUCTION

SIGN language is a type of visual communication process that incorporates hand gestures, body language and facial expressions. It is the main medium of communication for people who are not only deaf and hard-of-hearing people, but also other groups of people. Individuals with disabilities including mental imbalance, apraxia of discourse, cerebral paralysis, and down condition may likewise find communication via sign language advantageous for conveying.

This group of people have developed their language to communicate among themselves, known to us as Sign language. Sign languages use visual human body movements and gestures to express one's thoughts. Sign languages vary from region to region. For example, there is American Sign Language in the United States of America, whereas Indian Sign Language in India. There is also a part in different sign languages where native language alphabets are expressed by fingerspelling hand signs.

There is no single sign language that is used all over the world. Like spoken languages, sign languages too have been adapted naturally as people in different region used their own signs to communicate with one another, leading toa wide range of variations. There are somewhere close to 138 and 300 distinct kinds of sign language utilized all throughout the planet today. We have decided to focus on the American Sign Language (ASL). ASL is a complete, natural language that has similar etymological properties as communicated in dialects. It is the predominant language of many deaf and hard-of-hearing persons in North America. The fingerspelling of manual alphabets is the core of sign language. It is used to spell out proper nouns for which there are no signs. It can also be used to clarify a sign that is unfamiliar to the person who is trying to interpret it or to spell out words for signs that are unfamiliar to the person trying to communicate it. Fingerspelling signs are frequently used in conjunction with other ASL signs. ASL fingerspelling, unlike most sign languages, uses only one hand to sign alphabets. Our project classifies the 26 alphabets and 0 to 9 numeric values. These letters and gestures are recognized to form words and display the image of each word formed. The aim of this study is to make a system using deep learning to implement a real time sign language detection system. It should be able to read the frames of our video capture in real time, display the predictions and string them together to form words and later display the images for the words formed

The approach includes a methodology that involves implementation of a hand tracking solution provided by Google's open-source project, MediaPipe [6]. Along with it, a deep learning algorithm is implemented on this solution to produce a fast, inexpensive, lightweight, and easy-to-deploy system that can be used as the core of a complete sign language recognition system.

This work adopts the hand landmarks detection approach at the core of the complete model. Using MediaPipe's hand tracking model, hand landmarks in each image containing hand signs from American Sign Language have been detected. The landmarks are then collected as coordinate points, normalized, and saved in a .csv file as data-points. A feedforward neural network model than takes these data-points as input, and after training the model, real-time hand sign recognition with OpenCV has been implemented using the trained model. Fig. 1 gives an overview of the complete project. This paper discusses the entire procedure of developing our model, along with the results and analysis of performance.

II. EXISTING SYSTEM

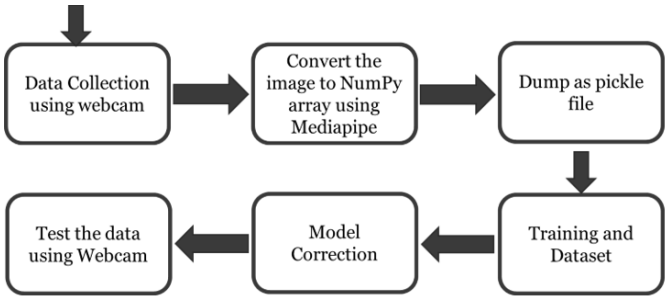
In the existing face recognition systems, Mittal et al. proposed a system that used an LSTM model with leap

motion for continuous sign language recognition. They employed a four-gated LSTM cell with a 2D CNN architecture for sign sentence recognition and assigned a particular label to every word. When the three basic gates were used as an input gate, a forget gate that took the output at the t-1time generated output at the output gate and returned the label showing the word that was specifically associated with that label. They added a special symbol, $, which indicated the transition between the two consecutive signs in the video. The RESET flag was used when they encountered the transition between two signs, which was indicated by the $ sign. They trained this modified LSTM model over a dataset and achieved a maximum accuracy of 72.3% using a 3-layer LSTM model for sign sentence recognition. For sign word recognition, the achieved accuracy was 89.50%.

III. PROPOSED SYSTEM

This work attempts to develop a technical approach for recognizing Sign Language, which is one of the 22 modern languages of India. Using machine learning techniques, this work tried to establish a system for identifying the hand gestures from Alphabetical Sign Language. A combination of two-dimensional and three-dimensional images of Alphabetical gestures has been used to prepare a dataset. The Media Pipe framework has been implemented to detect landmarks in the images. An Alphabetical Language dataset of 2094 data points has been generated, including nine static gestures with vowels and consonants from the Alphabetical Sign Language. The results reveal that the method implemented in this work is effective for the recognition of the other alphabets and gestures in the Sign Language. This method could also be tried and tested for the recognition of signs and gestures for various other local languages of India.

IV. SYSTEM ARCHITECTURE



1. Literature Review

Das et al. [5] have researched on deep learning-based sign language recognition system with processed static images implemented on American Sign Language gestures. The dataset consisted of 24 labels of static gestures of alphabets from A to Z, excluding J. There were approximately 100 images per class in the dataset captured on an RGB sensor. The methodology included using of Inception V3 convolutional neural network model to train the model. After training and testing the model, the average accuracy rate of validation was over 90%, with the most outstanding validation accuracy being 98%. They concluded that the relatively new Inception V3 model could be an appropriate model for static sign language detection when provided with a dataset of properly cropped images.

Sahoo [7] used machine learning to work on Indian sign language recognition. This research focused on the static hand gestures in Indian sign language for the numeric values (0-9). A digital RGB sensor was used to capture the images of the signs to build the dataset. The dataset contained 5000 total images, with 500 images for each digit from 0 to 9. Two classifiers were used based on the supervised learning technique to train the model: Naïve Bayes and k- Nearest Neighbor. K-Nearest Neighbor technique slightly performed better than the Naïve Bayes classifier in this research as the average accuracy rates of k-NN and Naïve Bayes were 98.36% and 97.79%, respectively.

Ansari and Harit [4] researched classifying Indian sign language static gestures using images with 3d depth data. The images were captured using Microsoft Kinect, which enables 3d depth information capturing along with the 2D image. The dataset totaled 5041 images of static hand gestures and was labeled with 140 classes. K-means clustering was used to train the model. In the research, they were able to score a 90% accuracy rate for 13 signs and 100% accuracy for three signs making it up to 16 alphabets (A, B, D, E, F, G, H, K, P, R, T, U, W, X, Y, Z) recognition with an average accuracy rate of 90.68%.

Rekha et al. [8] worked on 23 static and three dynamic signs of the Indian Sign Language dataset. They used skin color segmentation to locate hands. Edge orientation and texture were used as features to train a multiclass SVM on which they achieved a success rate of about 86.3%. However, their approach was too slow to implement as a practical gesture recognition algorithm.

Bhuyan et al. [9] used a dataset of 8 gestures from Indian Sign Language consisting of 400 images. They used a skin colour based segmentation technique for hand detection, then used nearest neighbor classification, and finally achieved a recognition rate above 90%.

Pugeault & Bowden [10] worked on a real-time recognition system for recognition of the alphabets in American Sign Language. A dataset consisting of 24 classes with 48,000 3D depth images captured using a Kinect sensor was used. Gabor filters and multi-class random forests were used and highly accurate classification rates were achieved.

Keskin et al. [11] used an approach involving object recognition by parts to recognize signs indicating digits of American Sign Language. Their dataset had ten classes with a total of 30,000 3d depth images captured using a Kinect sensor, and they achieved about 99% accuracy rate.

Ren et al. [8] used a threshold-based technique for segmentation of hands from Kinect captured images. They had ten classes with 1000 images in total in their dataset. The accuracy rate achieved was about 93%.

Halder and Tayade [12] used the MediaPipe framework to get multi-hand landmarks and used Support Vector Machine (SVM) for Real-time detection of hand signs. The average accuracy achieved was about 99%.

V. Data Collection Procedure

1.1 Dataset Collection

Data collection is a crucial step in developing a sign language. Here we collect data set using web camera. During Data collection we give the input in the format of images Collecting data for sign language recognition using a webcam involves capturing video footage of individuals performing sign language gestures. This data can then be used to train machine learning models for sign language recognition. Here are the steps you can follow to collect and prepare the data:

i. Hardware and Software Setup:

Use a webcam or a camera-equipped device to record video footage.

Choose appropriate lighting conditions to ensure clear video quality.

Install video recording software or use built-in webcam software on your computer.

ii. Data Collection Plan:

Define the set of sign language gestures or signs you want to recognize.

Plan the number of samples per gesture, depending on the complexity of the sign language and your available resources.

iii. Data Collection Procedure:

Invite individuals who are proficient in sign language to perform the selected gestures in front of the webcam. Record video sequences of each gesture from different angles and distances to create a diverse dataset. Ensure that each gesture is performed naturally and clearly to capture variations in hand shapes and movements.

iv. Annotation:

After recording, annotate the videos to label each frame or time segment with the corresponding sign gesture. You can use software tools designed for video annotation or manually label the data.

v. Data Preprocessing:

Extract individual frames from the video sequences to create a dataset of images. Resize and normalize the images to a consistent resolution to ensure uniformity in the dataset. Split the dataset into training, validation, and test sets for model development and evaluation.

vi. Data Augmentation:

To increase the diversity of your dataset, apply data augmentation techniques such as rotation, translation, scaling, and flipping to the images. Augmentation helps improve the model's robustness.

vii. Storage and Backup:

Store your dataset in a well-organized directory structure. Create backups of your dataset to prevent data loss.

viii. Privacy and Consent:

Obtain consent from individuals participating in the data collection, ensuring that they understand how their data will be used and that their privacy is protected.

ix. Model Development:

Use the collected and preprocessed data to train a sign language recognition model. Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) are commonly used architectures for this task.

x. Evaluation and Iteration:

Evaluate the model's performance using the validation and test sets. If the model's performance is unsatisfactory, consider collecting more data, fine-tuning the model, or improving the data preprocessing techniques.

xi. Deployment:

Once the model performs well, you can deploy it in applications that recognize sign language gestures in real-time using a webcam.

VI. Data Conversion

i. Detection of hand landmarks using MediaPipe

MediaPipe framework is used to build machine learning pipelines for time-series data such as video, audio, etc. Google first introduced it to perform real-time video and audio analysis on YouTube. In 2019, MediaPipe's public release enabled researchers and developers to integrate and use this framework in their projects. Unlike most high computing power demanding machine learning frameworks, MediaPipe can run efficiently, maintaining its accuracy and robustness on devices with low computing power, such as androids and embedded IoT devices. MediaPipe toolkit altogether consists of the MediaPipe framework and MediaPipe solutions. The MediaPipe framework is developed using C++, Java, and Objective C programming, which consists of 3 key APIs

- Calculator API
- Graph Construction API and
- Graph Execution API

MediaPipe solutions comprise 16 pre-trained TensorFlow and TensorFlow Lite models on top of the MediaPipe framework built for specific use cases. This work leveraged the MediaPipe solution to infer hand landmarks from images of hand signs. This hand-tracking model outputs 21 3D landmark points (as

shown in Fig. 1) on a hand from a single frame. To achieve this, two dependent models are used simultaneously. First, a Palm Detection Model detects the palms of hands in the images since it is easier to detect rigid objects like palms and fists rather than a complete hand. Cropped palm images from this model are passed onto the next model, which is the Hand Landmark Model. This model precisely detects 21 3D hand landmark points in the detected hand region using regression. The complete model is trained on approximately 30,000 manually annotated real-world images. The model is very well-trained and robust and hence it can detect and map hand landmark points accurately, even on partially visible hands in most cases.

Fig. 1. 21 3D hand landmarks localized by MediaPipe hand tracking model

The hand tracking model from MediaPipe was passed through all the collected images of hand signs from American sign language . The model output x, y, and z coordinates of the hand landmark points from the images, among which only x and y coordinates are necessary and sufficient for training the final model. Therefore, the z coordinates were eliminated, and the remaining coordinate points (x and y) of the hand landmarks for different hand signs are stored in a .csv file. Fig. 2 shows how the milestones are saved as coordinates. These coordinates were the data points using which the final dataset was generated and used to train the final model.

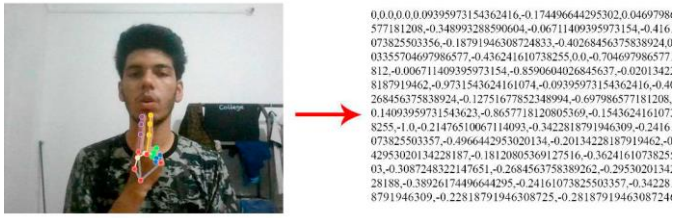


Fig. 2. Coordinates of the hand landmark points excluding z coordinates

ii. Data cleaning and normalization

The MediaPipe hand landmarks model gives the coordinates indicating hand landmark points based on the placement of the pixels containing the landmark points in the image. Therefore, the coordinates of two images of the same hand sign with different arrangements in the frame can be vastly distant. This increases the difficulty of training the model. To solve this problem, the wrist's landmark point has been considered (donated by index 0 in the hand landmarks list) to have x and y coordinates as (0,0) and accordingly adjusted the coordinates of all the other landmark points relative to the wrist point. Then we further normalized the coordinate values to be in the range of [0,1] by dividing all the coordinates by the largest absolute coordinate value obtained by relative adjustment in the landmarks list. After normalization of the coordinates, they were collected in the .csv file. Fig. 3 demonstrates the coordinate normalization procedure. After collecting coordinates in the .csv file, it was passed through a pandas' library function to detect null entries. Sometimes in blurry images, the model fails to detect hands, producing void entries. Cleaning of these void entries is necessary to train an unbiased model. Therefore, we detected the null entries and removed them from the file using their indices.

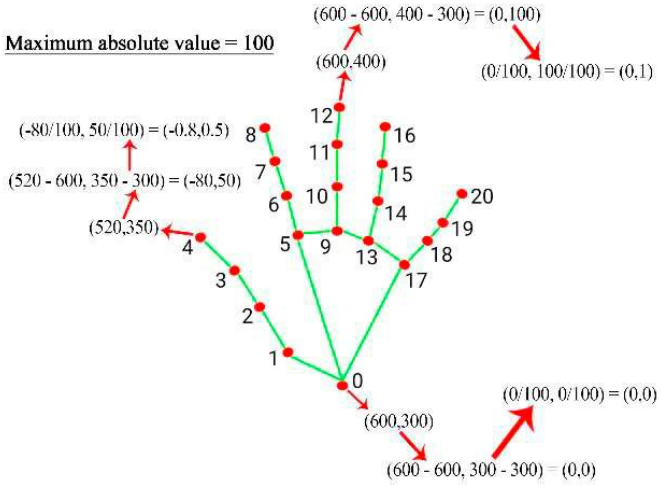


Fig. 3. Hand landmarks coordinates normalization example

2. Training the Model with Custom Feedforward Neural Network

After creating the dataset, the next task is the training of a feedforward neural network with the data. The neural network included one input layer and one output layer along with four hidden layers, including two dense layers and two dropout layers. The dense layers perform matrix-vector multiplication in the background, and the dropout layers reduce the overfitting of the model by modifying outgoing edges of hidden layer neurons randomly and resetting it to 0 at each iteration during the training process. A ReLU activation function was used in the hidden layers, and a Softmax activation function was used in the output layer. Table 1 presents the configuration of the sequential model implemented in this work. After construction, the model was compiled using an Adam optimizer which is computationally efficient and appropriate for model training with large data/parameters. The loss function was set to Sparse Categorical Crossentropy, which gives loss between actual and predicted labels. The accuracy metric was used for the evaluation of the model, which depicts how often the prediction is equal to the actual label. An early stopping function was used in training the model, which stops the training if there is not much variation in the calculated loss and accuracy in 20 consecutive training steps. Initially, the epoch value of the training was set to 1000, but the model's training was completed in 341 epochs and stopped using the early stopping function. Training accuracy obtained was 99.40%, and the calculated loss was 0.1090.

Table 1: Configuration of the used feedforward neural network

Model: "sequential"		
Layer (type)	**Output Shape**	**No of Parameters**
dropout (Dropout)	(None, 42)	0
dense (Dense)	(None, 20)	860
dropout_1 (Dropout)	(None, 20)	0
dense_1 (Dense)	(None, 10)	210
dense_2 (Dense)	(None, 10)	110
Total parameters: 1,180 Trainable parameters: 1,180 Non- trainable parameters: 0		

3. Results and Analysis

To test the data open the webcam and give sign in your fingers as input it will predict the output as alphabetical letters.

VII. CONCLUSION

This work attempts to provide a visual solution for the sign language recognition problem in the regional Indian language, where different sets of alphabets exist for each. A state-of-the-art methodology has been adapted to implement the solution using advanced tools like MidiaPipe. Both real-time and static gestures have been tried to be recognized by training a self-generated 3D and 2D image dataset. The classification results claim that compared to other

models in the literature, which require high computation power and longer training time, this approach of sign language recognition using the MediaPipe hand tracking solution is more effective and faster for classifying complex hand signs and gestures including alphabets. Moreover, the implementation of MediaPipe also ensures accurate tracking of the hand movements with different motions in the finger phalanges and finger joint deviations. Also, due to its lightweight characteristic, the model becomes more robust and can be implemented over various computing devices with different computing power without losing speed and accuracy. This work can be further extended to include recognition of more signs from the Sign Language, including dynamic gestures involved in daily life communications. Also, various deep learning techniques can be tested after implementing MediaPipe's hand tracking solution to increase the accuracy and efficiency of the model

VIII. REFERENCES

[1]. A. Z. Shukor, M. F. Miskon, M. H. Jamaluddin, F. bin Ali, M. F. Asyraf, M. B. bin Bahar and others. (2015) "A new data glove approach for Malaysian sign language detection," Procedia Computer Science, vol. 76, p. 60–67.

[2]. M. Mohandes, M. Deriche and J. Liu. (2014) "Image-based and sensor-based approaches to Arabic sign language recognition," IEEE transactions on human-machine systems, vol. 44, p. 551–557.

[3]. N. M. Kakoty and M. D. Sharma. (2018) "Recognition of sign language alphabets and numbers based on hand kinematics using a data glove," Procedia Computer Science, vol. 133, p. 55–62.

[4]. Z. A. Ansari and G. Harit. (2016) "Nearest neighbour classification of Indian sign language gestures using kinect camera," Sadhana, vol. 41, p. 161–182.

[5]. A. Das, S. Gawde, K. Suratwala and D. Kalbande. (2018) "Sign language recognition using deep learning on custom processed static gesture images," in International Conference on Smart City and Emerging Technology (ICSCET)

[6]. C. Lugaresi, J. Tang, H. Nash, C. McClanahan, E. Uboweja, M. Hays, F. Zhang, C.-L. Chang, M. G. Yong, J. Lee and others. (2019) "Mediapipe: A framework for building perception pipelines," arXiv preprint arXiv:1906.08172.

[7]. A. K. Sahoo. (2021) "Indian sign language recognition using machine learning techniques," in Macromolecular Symposia.

[8]. J. Rekha, J. Bhattacharya and S. Majumder. (2011) "Shape, texture and local movement hand gesture features for indian sign language recognition," in 3rd international conference on trendz in information sciences & computing (TISC2011).

[9]. M. K. Bhuyan, M. K. Kar and D. R. Neog. (2011) "Hand pose identification from monocular image for sign language recognition," in 2011 IEEE International Conference on Signal and Image Processing Applications (ICSIPA).

[10]. N. Pugeault and R. Bowden. (2011) "Spelling it out: Real-time ASL fingerspelling recognition," in 2011 IEEE International conference on computer vision workshops (ICCV workshops).

[11]. C. Keskin, F. Kıraç, Y. E. Kara and L. Akarun. (2013) "Real time hand pose estimation using depth sensors," in Consumer depth cameras for computer vision, Springer, p. 119–137.

[12]. A. Halder and A. Tayad. (2021) "Real-time vernacular sign language recognition using mediapipe and machine learning," Journal homepage: www. ijrpr. com ISSN, vol. 2582, p. 7421.

[13]. D. A. Kumar, A. S. C. S. Sastry, P. V. V. Kishore and E. K. Kumar. (2022) "3D sign language recognition using spatio temporal graph kernels," Journal of King Saud University-Computer and Information Sciences.

[14]. Z. Ren, J. Yuan and Z. Zhang. (2011) "Robust hand gesture recognition based on finger-earth mover's distance with a commodity depth camera," in Proceedings of the 19th ACM international conference on Multimedia.

Cite this article as :

Ms. E J Honesty Praiselin, Dr G Manikandan, Ms. Vilma Veronica, Ms. S. Hemalatha, "Sign Language Detection and Recognition Using Media Pipe and Deep Learning Algorithm", International Journal of Scientific Research in Science and Technology (IJSRST), Online ISSN : 2395-602X, Print ISSN : 2395-6011, Volume 11 Issue 2, pp. 123-130, March-April 2024. Available at doi : https://doi.org/10.32628/IJSRST52411223
Journal URL : https://ijsrst.com/IJSRST52411223