WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Real-Time Hand Gesture Recognition System Using MediaPipe and LSTM

[A]Sachin Agrawal, [B]Agnishrota Chakraborty, [C]M. Rajalakshmi *

Student,SRM Institute of Science and Technology, Chennai 603203, India
Student,SRM Institute of Science and Technology, Chennai 603203, India
Assistant Professor,SRM Institute of Science and Technology, Chennai 603203, India

A B S T R A C T

In the field of Human Computer Interaction (HCI), a hand gesture recognition system provides an innovative approach towards nonverbal communication with your machines. The objective is to overcome the communication barrier between the people with disabilities and those who do not understand sign language. The goal of this research is to present and develop a technique for hand gesture recognition that incorporates MediaPipe for extracting hand landmarks and LSTM to train and recognize the gesture. The setup comprises a web camera that captures the user's gesture and feeds it into the proposed model as an input. The proposed model will be implemented in following phases: Data collection, preprocessing training and testing of the proposed neural network, at last testing in real time. The Gesture recognition model is trained on a self-made dynamic dataset of certain everyday gestures. The trained model recognizes the gesture accurately and shows the gesture made in the form of text onto the screen.

Keywords: hand gesture, sign language, real time detection, lstm, mediapipe

1. INTRODUCTION

Human Computer Interaction (HCI) has seen advancement and improvements since its development but still major everyday interaction with machines is limited to fingers and eyes, whereas some of the parts of our body like legs, arms and mouth are underused or not used for interaction at all. Hand gesture Recognition system is a way for humans to interact with machines/computers by simply using our hand gestures. Hand gestures can add another dimension to communication with computers. The main purpose of this real-time hand gesture recognition system is to detect and recognize hand movements in real time. Hand recognition is a method that employs algorithms and neural networks to figure out how a hand moves and then recognize patterns of the same action. It has a broad array of applications including in the field of Automotive, Consumer Electronics, Healthcare industry, Entertainment- VR experience. The goal is to develop a Hand gesture Recognition system which is independent of external hardware variables or limitations such as any controller/gloves, high definition camera, and depth camera. The only requirement is a working camera which would take input and feed to the model to recognize the output pattern.

2. LITERATURE SURVEY

A Study [1] provided the knowledge of different approaches and techniques used for hand gesture recognition. It helped understanding Color Models, Image Segmentation & Image Histogram. Another approach [2] introduced Static and dynamic hand gesture, Skin Detection, Various applications of the technology (sign language, robotics, augmented reality games etc.).Limitations of lighting or background condition, which means that users have to be in a particular place while running the system. VGG-11 and VGG-16 have also been adapted and implemented for Hand Gesture Recognition, and a huge

* *Corresponding author.* Tel.:+91 9571566754; fax: +0-000-000-0000.

E-mail address: sachinagrawal769@gmail.com

collection of static sign language hand gestures has been developed. Results for this model [3] are obtained by the proposed work over the existing models. The approach [4] in which Using a deep convolutional neural network (D-CNN) and micro-Doppler signatures obtained by Doppler radar after five-fold validation, the classification accuracy of the proposed method was found to be 85.6%. Out of all, For only seven gestures, the accuracy increased to 93.1%. Limitation in providing balanced accuracy. Deep learning model [5] in which image processing and neural networks were used to create this. The majority of models produce identical test results for both plain and busy backgrounds. Limitation in removing the ambiguity produced in the results by introducing background fluctuation. Furthermore, they are unable to generate any signal from static hand gestures. Vision-based approaches, on the other hand, employ a single or several cameras as well as precise body motion sensors like Leap Motion controllers and Microsoft Kinect. A combination of CNN and LSTM [6] approaches have been previously used where the CNN is used to learn spatial features and gestures and the output is fed into LSTM to extract the temporal features. Similarly a 3D CNN and LSTM were used with a FSM for the contextual classification [7].

3. PROPOSED WORK

The proposed work takes into consideration the challenges faced by previous models and work to minimize them. We built a system that doesn't compromise between efficiency and performance. One of the major challenges faced by scholars was the non-uniform background and segmentation of hands from the background. We eliminated that problem by creating a dataset using Google's Mediapipe solution and openCV library to detect the landmarks from the hand and also used the same during the real time detection part of the hand gesture recognition system so it doesn't matter whether one is in their car, at home or in streets, the system will detect accurate landmarks from one's hands. The proposed system has three main modules connected in sequence as shown in fig. The Dataset module where the landmarks are extracted and the dataset creation process is done followed by the preprocessing module where the data is processed to feed into the final LSTM module in which the training for recognizing the gestures takes place. Finally when the model is producing satisfactory results after tuning the hyper parameters, a camera renders the live real time feed and passes it through the model and the name of the gesture is shown as text onto the screen. Figure 1 shows the architecture of the proposed Hand Gesture Recognition System.

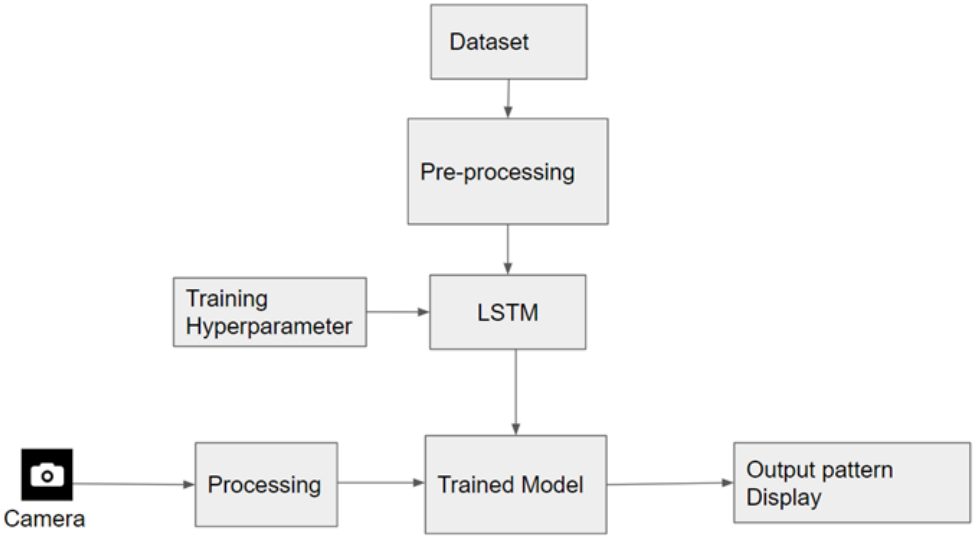

Fig. 1 - Architecture of the Hand Gesture Recognition System

4. MODULES AND IMPLEMENTATION

4.1 Hand Detection using Mediapipe Holistic model

Mediapipe as itself offers a wide range of solutions, within the holistic pipeline of mediapipe, it consists of three components: pose, hand and face. The holistic model extracts a sum of 543 individual landmarks (468- face, 33- pose and 21- hand). We are using Mediapipe holistic model included in mediapipe python module. We defined a function (mediapipe_detection) that takes two arguments 1. Image: The image in which it performs detection, 2. Model: The mediapipe model that we are using for detection ('Holistic model'). The function converts the image from BGR to RGB format, the image is then passed through model.process() function and the result is stored. The function then converts the image back to BGR format, and then returns the image and the stored result. We define another function (draw_styled_landmarks), it takes the returned output from the previous functions as its arguments. Using this function we draw the landmarks, which helps visualize the real-time hand detection. Extract Key-points: After real-time hand detection is successfully achieved, we then extract the X,Y and Z coordinates of the key points from the detection results(the stored result returned by the

mediapipe_detection function mentioned above). We define another method (extract_keypoints) that takes the detection results as its argument, to perform the extraction of the coordinates. This function returns the concatenated array of all the arrays containing the key point coordinates of the holistic model. This function is called during data collection. While we used the holistic model and extracted key points for all three: hand, face and pose, we only showed hand landmarks (see Figure 2) on the screen during the recognition phase to avoid clusters of landmarks on the screen.

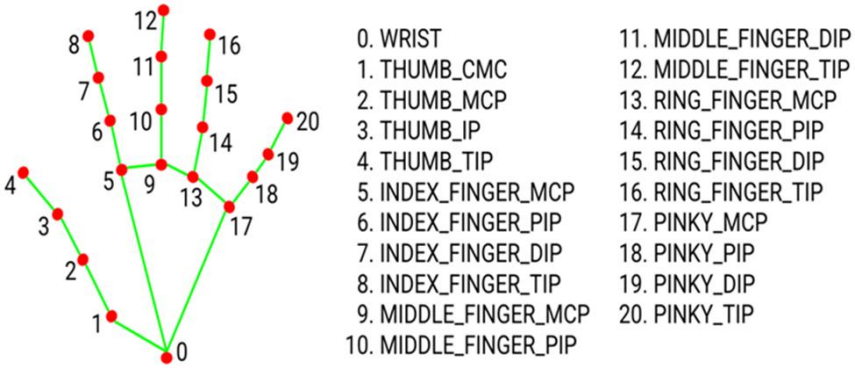

Fig. 2 - The hand_landmarks model of Mediapipe

4.2 Data Collection

The first step is to set up folders for data collection-We have collected our own data-set for the training and testing of our deep learning model in the following way: We created a folder for the dataset using: os.path.join ('data_folder_name') method. We then defined an array of 10 gestures (['hello', 'thanks', 'i love you', 'stop', 'please', 'walk', 'argue', 'yes', 'see', 'good']) that will be used to train our model for recognition. We define the number of videos (80) we are going to collect and the number of frames (30) for each video to be captured. After we are done with creation of the gesture folder, the next step is collection of data in folders-A loop is then run to create the 80 video folders for each of the gestures. E.g. For each of the gesture folder there are video folders as shown in the Figure 3 below:

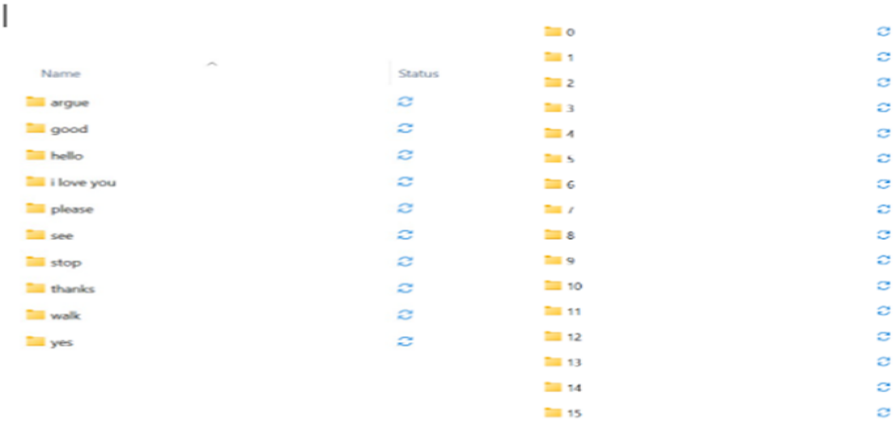

Fig. 3 - Folder for gestures and corresponding subfolders for each gesture

Then the next step is to collect the video data for each gesture and use the extract_keypoints function to collect the array of key points for each frame. Finally the numpy array of key points collected is saved for each frame inside each video folder as shown in the Figure 4 below, in the figure it shows till 10.npy it actually goes till 29.npy.

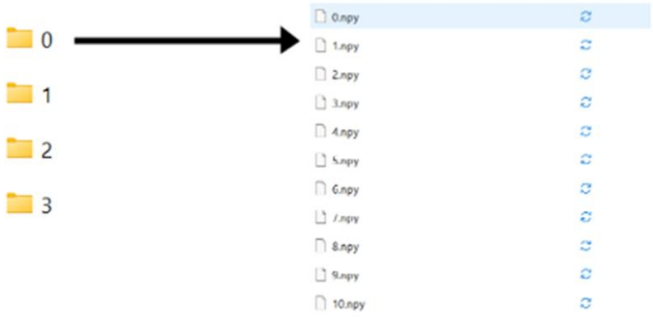

Fig. 4 - Collection of 30 frames inside each subfolder

4.3 Data Preparation and Label Creation

After the data collection process is complete, the key points extracted from the data is then structured using data preprocessing, the Data is structured such a way that all the arrays of key points of each gesture is saved as one numpy array (X) which is then mapped to another numpy array of labels(Y). We then use the to_categorical function to convert Y into a binary class matrix e.g. [1,0,0….] represents hello, [0,1,0,0….] represents thanks and [0,0,1,0,0,.....] represents "i love you" and so on. After the preprocessing is complete the data is then split into training and testing data using train_test_split function.

4.4 Implementation of LSTM

Long short term memory (LSTM) is a variation of RNN which is mainly used for sequence of data or data in time-series. The LSTM unit is more complex than a RNN unit. It has gates that control the flow of information from an individual unit. The loop is a crucial component of the LSTM architecture since it has the ability to retain previous history of information in the internal state memory. This is done to identify both spatial and temporal features from data.

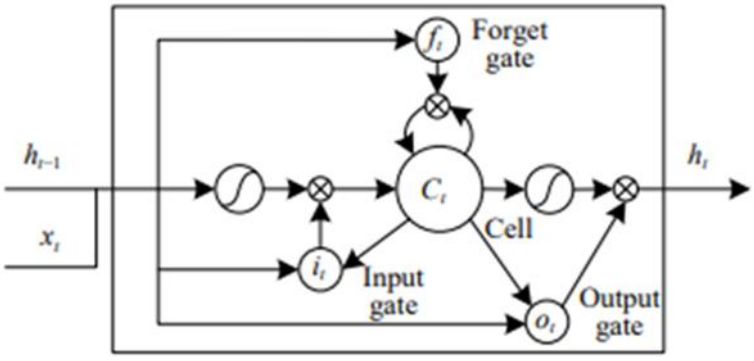

Fig. 5 - Architecture of a LSTM unit

The architecture of a single LSTM model can be understood as having three gates and a module called a memory cell (see Figure 5 & Figure 6). Forget Gate- The forget gate removes information from the cell state that is no longer helpful. The forget gate states that what kind of information can be forgotten and is no longer contextually relevant. Input Gate- Work of it is to add important information to the cell state. It deals that what new information we should add or update into our working storage state information. Output Gate- It is in charge of retrieving valuable information from the current cell state and presenting it as output. Out of all the information which is stored in the current state, which part of it should be given as the output in a particular

instance. Memory Cell- The memory cell has 2 states: short term memory and long term memory which is a self-state. Each gate is assigned values between 0-1 on the working of that gate. If its 0 no information is passed and 1 is all information and also the values between determines the information passed. The module of a single LSTM here has 4 layers which interact.

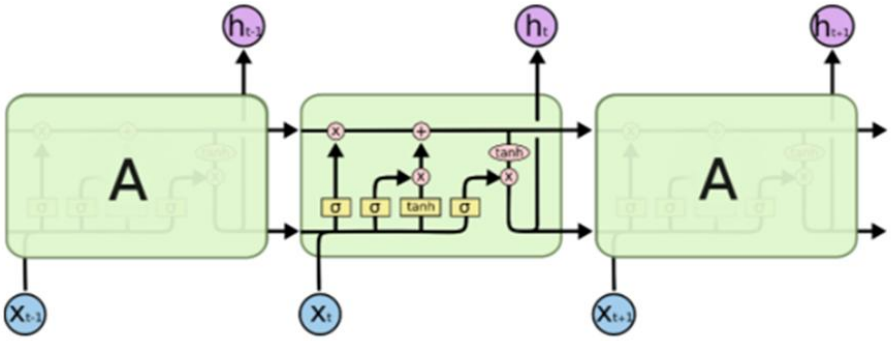

Fig. 6 -An LSTM module over time

Our data consists of a sequence of frames stored together where each frame has information regarding the position of various extracted landmarks. Each of these frames is stored as a numpy array. The lstm network is built with the sequential model. There are 4 layers for LSTM followed by three dense layers. 'ReLu' activation function is used for the first 6 layers and 'softmax' function for the final dense layer. The optimizer used is ADAM to deal with stochastic gradient descent. During the training of the model, both training and validation accuracy and loss is measured after each epoch for evaluation.

5. RESULTS AND DISCUSSION

5.1 Testing on validation data

The dataset used for testing was taken from the original dataset where it was divided between training and testing data. The evaluation metrics we use is Accuracy. The LSTM model is working successfully(see figure 7). Accuracy of 90% was achieved with the test dataset

```
res = model.predict(X_test)

actions[np.argmax(res[4])]
'yes'

actions[np.argmax(y_test[4])]
'yes'
```

Fig. 7 -Comparison between test data and predicted result

5.2 Testing in real time

Using live feed, the recognition in the form of text was shown for gestures (see figure 8). Real time testing was done on each gesture with promising results. As our model is trained to perform the prediction on 30 frames worth of key points, thus in order to enable it in real-time the video feed is processed in the following manner: An empty array: sequence = [ ] is defined. The real-time feed is captured using open CV, and various methods that have been explained earlier in this paper such as: mediapipe_detection and extract_keypoints are called upon to perform the detection and collect the key points in real time. The collected key points are appended to the empty array we defined in the beginning. The array is then assigned as: sequence = [-

30:]; Thus, it saves the last 30 frames worth of key points appended, When the length of the sequence array is determined as 30, then the model is called upon to predict the output.

Fig. 8 -Recognition of few gestures in real time

6. CONCLUSION AND FUTURE ENHANCEMENTS

This complete study proposed and developed a system for hand gesture recognition using trendy MediaPipe and LSTM. Initially a folder was created for each of 10 gestures and for each gesture 80 subfolder was created, those subfolders can be considered as video folder and inside each subfolder lies 30 frames where each frame is in the form of numpy array containing landmarks values detected and extracted using Mediapipe Holistic Solution. The LSTM network was trained using the data and gave accuracy of 90% on testing data. Finally the system was tested using real time data directly fed into the model and the result was shown on the screen for each gesture. Some lag was seen while recognizing the gestures in real time. Future works are as follows: Add a lot more everyday dynamic gestures and more data for each of those gestures. Currently the model is trained on words; it can be further taken and trained on complete sentences. Deploy the complete system on the web so that any user can make use of their camera to simply recognize the gesture. Making the system compatible with the mobile device.

REFERENCES

Anju S R, SubuSurendran, 2014, A Study on Different Hand Gesture Recognition Techniques, INTERNATIONAL JOURNAL OF ENGINEERINGRESEARCH & TECHNOLOGY (IJERT) Volume 03, Issue 04 (April 2014).https://www.ijert.org/a-study-on-different-hand-gesture-recognition-techniques.

Rahmat, Romi&Chairunnisa, T. &Gunawan, Dani& Pasha, Muhammad Fermi &Budiarto, Rahmat.(2019). Hand gestures recognition with improved skin color segmentation in human-computer interaction applications.Journal of Theoretical andAppliedInformationTechnology2019.

Sakshi Sharma, Sukhwinder Singh, "Vision-based hand gesture recognition using deep learning for the interpretation of sign language", Expert Systems with Applications: An International Journal Volume 182 Issue C Nov 2021.https://doi.org/10.1016/j.eswa.2021.115657.

Y. Kim and B. Toomajian, "Hand Gesture Recognition Using Micro-Doppler Signatures With Convolutional Neural Network," in IEEE Access, 2016.doi: 10.1109/ACCESS.2016.2617282.

Agrawal, M., Ainapure, R., Agrawal, S., Bhosale, S. & Desai, S. S. (2020). Models for Hand Gesture Recognition using Deep Learning. 2020 IEEE 5th International Conference on Computing Communication and Automation, 2020,pp. 589-594, doi: 10.1109/ICCCA49541.2020.9250846.

Jiang, S., Chen, Y. (2018). Hand Gesture Recognition by Using 3DCNN and LSTM with Adam Optimizer. In: Zeng, B., Huang, Q., El Saddik, A., Li, H., Jiang, S., Fan, X. (eds) Advances in Multimedia Information Processing – PCM 2017. PCM 2017.https://doi.org/10.1007/978-3-319-77380-3_71 NoorkholisLuthfil Hakim, Timothy K. Shih , SandeliPriyanwadaKasthuriArachchi, WisnuAditya 1, Yi-Cheng Chen and Chih-Yang Lin, "Dynamic Hand Gesture Recognition Using 3DCNN and LSTM with FSM Context-Aware Model" 2019. https://doi.org/10.3390/s19245429.

Wenjin Zhang, Jiacun Wang, Senior Member, IEEE, and FangpingLan,"Dynamic Hand Gesture Recognition Based on Short-Term Sampling Neural Networks" in IEEE/CAA JOURNAL OF AUTOMATICA SINICA ,2021 https://ieeexplore.ieee.org/document/9272702

HeyuanGuo, Yang Yang, and HuaCai, "Exploiting LSTM-RNNs and 3D Skeleton Features for Hand Gesture Recognition" in Proceedings of the 2nd WRC

Symposium on Advanced Robotics and Automation, 2019. https://ieeexplore.ieee.org/document/8931937

Y. Wu, B. Zheng and Y. Zhao, "Dynamic Gesture Recognition Based on LSTM-CNN," 2018 Chinese Automation Congress (CAC), 2018,pp. 2446-2450, doi: 10.1109/CAC.2018.8623035.

V. Sharma, M. Jaiswal, A. Sharma, S. Saini and R. Tomar, "Dynamic Two Hand Gesture Recognition using CNN-LSTM based networks," 2021 IEEE International Symposium on Smart Electronic Systems (iSES), 2021,pp. 224-229, doi: 10.1109/iSES52644.2021.00059. https://ieeexplore.ieee.org/document/9701135

Karpathy, A., Toderici, G., Shetty, S., et al.: Large-scale video classification with convolutional neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,2014 pp. 1725–1732,doi: 10.1109/CVPR.2014.223. https://ieeexplore.ieee.org/document/6909619

Basnin, N., Nahar, L., Hossain, M.S. (2021). An Integrated CNN-LSTM Model for Micro Hand Gesture Recognition. In: Vasant, P., Zelinka, I., Weber, GW. (eds) Intelligent Computing and Optimization.ICO 2020.Advances in Intelligent Systems and Computing,vol1324.Springer,Cham,2021.https://www.researchgate.net/publication/349148269_An_Integrated_CNN-LSTM_Model_for_Micro_Hand_Gesture_Recognition.