

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/369945035>

Real-time Vernacular Sign Language Recognition using MediaPipe and Machine Learning

Research · May 2021

DOI: 10.13140/RG.2.2.32364.03203

CITATIONS

41

READS

1,033

2 authors, including:



[Akshit Tayade](#)

Stevens Institute of Technology

3 PUBLICATIONS 44 CITATIONS

SEE PROFILE



Real-time Vernacular Sign Language Recognition using MediaPipe and Machine Learning

Arpita Halder^a, Akshit Tayade^b

^aUndergraduate Student, Computer Science and Engineering Department, Budge-Budge Institute of Technology, arpitahalder739@gmail.com, India

^bUndergraduate Student, Electronics and Telecommunication Department, K.J Somaiya College of Engineering, tayadeakshit28@yahoo.com, India

ABSTRACT

The deaf-mute community have undeniable communication problems in their daily life. Recent developments in artificial intelligence tear down this communication barrier. The main purpose of this paper is to demonstrate a methodology that simplified Sign Language Recognition using MediaPipe's open-source framework and machine learning algorithm. The predictive model is lightweight and adaptable to smart devices. Multiple sign language datasets such as American, Indian, Italian and Turkey are used for training purpose to analyze the capability of the framework. With an average accuracy of 99%, the proposed model is efficient, precise and robust. Real-time accurate detection using Support Vector Machine (SVM) algorithm without any wearable sensors makes use of this technology more comfortable and easy.

Keywords: Machine Learning, Sign Language Recognition, MediaPipe, Feature extraction, Hand gesture

1. Introduction

Sign Language significantly facilitates communication in the deaf community. Sign language is a language in which communication is based on visual sign patterns to express one's feelings. There is a communication gap when a deaf community wants to express their views, thought of speech and hearing with normal people. Currently, two communities mostly rely on human-based translator which can be expensive and inconvenient. With the development in areas of deep learning and computer vision, researchers have developed various automatic sign language recognition methods that can interpret sign gestures in an understandable way. This narrow downs the communication gap between impaired and normal people. This also empowers deaf-mute people to stand with an equal opportunity and improve personal growth.

In accordance with the report of the World Federation of the Deaf (WFD) over 5% of the world's population (\approx 360 million people) has hearing impairment including 328 million adults and 32 Million children. Approximately there are about 300 sign language is in use around the globe. Sign language recognition is a challenging task as sign language alphabets are different for different sign languages. For instance, American Sign Language (ASL) alphabets vary widely from Indian Sign Language or Italian Sign Language. Thus Sign language varies from region to region. Moreover, articulation of single as well as double hands is used to convey meaningful messages. Sign Language can be expressed by the compressed version, where a single gesture is sufficient to describe a word. Now, sign language also has fingerspelling to describe each alphabet of the word using different signs corresponding to a particular letter. As there are many words still not standardized in sign language dictionaries, fingerspelling is often used to manifest a word. There are still about 150,000 words in spoken English having no counterpart in ASL. Furthermore, any name of people, places, brands or titles doesn't have any standardized sign symbol. Besides, a user might not be aware of the exact sign of any particular word and in this scenario, fingerspelling comes in handy and any word can be easily described.

** Corresponding author.*

E-mail address: arpitahalder739@gmail.com

Previous works included sensor-based Sign language Recognition (SLR) system, which was quite uncomfortable and more restrictive for signers. Specialized hardware for example sensors [1], [2] were used which were an expensive option as well. Whereas, computer vision-based techniques uses bare hands without any sensors or coloured gloves. Due to the use of single camera, computer-vision based technique is more cost-effective and highly portable compared to sensor-based techniques. In computer-vision based methods, the most common approach for hand-tracking is skin colour detection or background subtraction. Computer vision-based SLR system often deals with feature extraction example boundary modelling, contour, segmentation of gestures and estimation of hand shapes. But, all these solutions are not lightweight enough to run in real-time devices like mobile phone applications and thus are restricted to platform equipped with robust processors. Moreover, the challenge of hand-tracking remained persistent in all these techniques. To address this drawback, our proposed methodology used an approach that involves Google's innovative, rapidly growing and open source project MediaPipe and a machine learning algorithm on top of this framework to get a faster, simpler, cost-effective, portable and easy to deploy pipeline which can be used as a sign language recognition system.

2. Related Works


Relatively hand gesture recognition is a difficult problem to address in the field of machine learning. Classification methods can be divided into supervised and unsupervised method. Based on these methods the SLR system can recognize static or dynamic sign gestures of hands. Murakami and Taguchi [3] in the year 1991, published a research article using neural network for the first time in sign language recognition. With the development in the field of computer vision, numerous researchers came up with novel approaches to help the physically challenged community. Using coloured gloves, a real-time hand tracking application was developed by Wang and Popovic[4]. The colour pattern of the gloves was recognized by K-Nearest Neighbors (KNN) technique but continuous feeding of hand streams is required for the system. However, Support Vector Mechanism (SVM) outperformed this algorithm in the research findings of Rekha et al.[5], Kurdyumov et al.[6], Tharwat et al.[7] and Baranwal and Nandi[8]. There are two types of Sign Language Recognition: Isolated sign recognition and continuous sentence recognition. Likewise, whole sign level modelling and subunit sign level modelling exist in the SLR system. Visual-descriptive and linguistic-oriented are two approaches that lead to subunit level sign modelling. Elakkiya et al.[9] combined SVM learning and boosting algorithm to propose a framework for subunit recognition of alphabets. An accuracy of 97.6% was obtained but the system fails to predict 26 alphabets. To extract features of 23 isolated Arabic sign language Ahmed and Aly[10] used the combination of PCA and local binary patterns. Despite getting an accuracy of 99.97% in signer dependent mode, due to the usage of threshold operator the system fails to recognize the constant grey-scale patterns in the signing area. In the field of machine learning, recognizing hand gesture is relatively problematic to solve. In most of the initial attempts, a conventional convolutional network is used that detects handgestures from frames of images. R.Sharma et al.,[11] used 80000 individual numeric signs with more than 500 pictures per sign to train a machine learning model. Their system methodology comprises a training database of pre-processed images for a hand-detection system and a gesture recognition system. Image pre-processing included feature extraction to normalize the input information before training the machine learning model. The images are converted into grayscale for better object contour maintaining a standardized resolution and then flattened into a smaller amount of one-dimensional components. The feature extraction technique helps to extract certain features about the pixel data from images and feed them to CNN for easier training and more accurate prediction. Hand tracking in 2D and 3D space has been performed by W.Liu et al.,[12].They used skin saliency where skin tones within a specific range were extracted for better feature extraction and achieved a classification accuracy of around 98%.

It is evident from all these previous methods that to recognize hand gesture precisely with high accuracy, models require a large dataset and complicated methodology with complex mathematical processing. Pre-processing of images plays a vital in the gesture tracking process. Therefore, for our project, we used an open-source framework from Google known as Mediapipe which is capable of detecting human body part accurately.

3. Dataset

Table 1: Details of different sign language fingerspelling datasets used in this work

Database	Type	No. of classes	No. of images	Image Samples
American	Alphabets	26	156000	
Indian	Alphabets	24	4972	

Italian	Alphabets	22	12856	
American	Numbers	10	1400	
Turkey	Numbers	10	4124	

4. Architecture

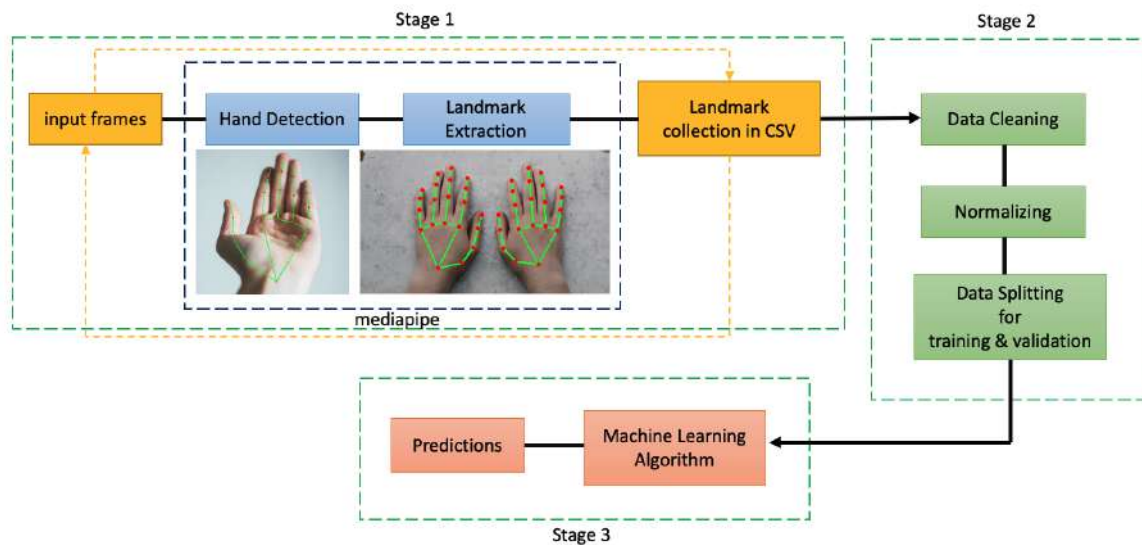


Figure 1: Proposed architecture to detect handgestures and predict sign language finger-spellings

1.1 Stage 1: Pre-Processing of Images to get Multi-hand Landmarks using MediaPipe

MediaPipe is a framework that enables developers for building multi-modal(video, audio, any times series data) cross-platform applied ML pipelines. MediaPipe has a large collection of human body detection and tracking models which are trained on a massive and most diverse dataset of Google. As the skeleton of nodes and edges or landmarks, they track key points on different parts of the body. All co-ordinate points are three-dimension normalized. Models build by Google developers using Tensorflow lite facilitates the flow of information easily adaptable and modifiable via graphs. MediaPipe pipelines are composed of nodes on a graph which are generally specified in ptxt file. These nodes are connected to C++ files. Expansion upon these files is the base calculator class in MediaPipe. Just like a video stream this class gets contracts of media streams from other nodes in the graph and ensures that

it is connected. Once, rest of the pipelines nodes are connected, the class generates its own output processed data. Packet objects encapsulating many different types of information are used to send each stream of information to each calculator. Into a graph, side packets can also be imposed, where a calculator node can be introduced with auxiliary data like constants or static properties. This simplified structure in the pipeline of dataflow enables additions or modifications with ease and the flow of data becomes more precisely controllable.

The Hand tracking solution [13] has an ML pipeline at its backend consisting of two models working dependently with each other: a) Palm Detection Model b) Land Landmark Model. The Palm Detection Model provides an accurately cropped palm image and further is passed on to the landmark model. This process diminishes the use of data augmentation (i.e. Rotations, Flipping, Scaling) that is done in Deep Learning models and dedicates most of its power for landmark localization. The traditional way is to detect the hand from the frame and then do landmark localization over the current frame. But in this Palm Detector using ML pipeline challenges with a different strategy. Detecting hands is a complex procedure as you have to perform image processing and thresholding and work with a variety of hand sizes which leads to consumption of time. Instead of directly detecting hand from the current frame, first, the Palm detector is trained which estimates bounding boxes around the rigid objects like palm and fists which is simpler than detecting hands with coupled fingers. Secondly, an encoder-decoder is used as an extractor for bigger scene context.



Figure 2: 21 Hand Landmarks

After the palm detection is skimmed over the whole image frame, subsequent Hand Landmark models comes into the picture. This model precisely localize 21 3D hand-knuckle coordinates (i.e., x, y, z-axis) inside the detected hand regions. The model is so well trained and robust in hand detection that it even maps coordinates to partially visible hand. Figure 2 shows the 21 landmark points detection by the Hand Landmark model.

Now that we have a functional Palm and Hand detection model running, this model is passed over our dataset of various language. Considering the American Sign Language dataset, we have a to z alphabets. So, we pass our detection model over every alphabet folder containing images and perform Hand detection which yields us the 21 landmark points as shown in Figure 2. The obtained landmark points are then stored in a file of CSV format. A simultaneous, elimination task is performed while extracting the landmark points. Here, only the x and y coordinates detected by the Hand Landmark model is considered for training the ML model. Depending upon the size of the dataset around 10-15 minutes is required for Landmark extraction.

1.2 Stage 2: Data cleaning and normalization

As in stage 1, we are only considering x and y coordinates from the detector, each image in the dataset is passed through stage 1 to collect all the data points under one file. This file is then scraped through the pandas' library function to check for any nulls entries. Sometimes due to blurry image, the detector cannot detect the hand which leads to null entry into the dataset. Hence, it is necessary to clean these points or will lead to biasness while making the predictive model. Rows containing these null entries are searched and using their indexes removed from the table. After the removal of unwanted points, we normalized x and y coordinates to fit into our system. The data file is then prepared for splitting into training and validation set. 80% of the data is retained for training our model with various optimization and loss function, whereas 20% of data is reserved for validating the model.

1.3 Stage 3: Prediction using Machine Learning Algorithm

Predictive analysis of different sign languages are performed using machine learning algorithms and Support Vector Machine (SVM) outperformed other algorithms. The details of the analysis are discussed in table 2 in the result section. SVM is effective in high dimensional spaces. In the case where the number of samples are greater than the number of dimensions, SVM performs effectively. SVM is a cluster of supervised learning methods capable of classification, regression and outliers detection.

The following formula poses the optimization problem tackled by SVMs:

$$\min(w, b, d) \frac{1}{2} w^T w + C \sum_{i=1}^n d_i \quad (1)$$

$$y_i (w^T \phi(x_i) + b) > 1 - d_i \quad (2)$$

In equation (1) and equation (2), d_i denotes the distances to the correct margin with $d_i \geq 0$, $i = 1, \dots, n$, C denotes a regularization parameter, $w^T w = \|w\|^2$ denotes the normal vector, $\phi(x_i)$ denotes the transformed input space vector, b denotes a bias parameter, y_i denotes the i -th target value. The objective is to classify as many data points correctly as possible by maximizing the margin from the Support Vectors to the hyperplane while minimizing the term $w^T w$. The kernel function used is RBF (radical basis function) that turns the input space into a higher-dimensional space, so that not every data point is explicitly mapped. SVM works relatively well when there is a clear margin of separation between classes. Hence, we used SVM to classify multiple classes of sign language alphabets and numerics.

1.4 Quantitative Analysis

To analyze results for each of the datasets, we used performance matrix such as accuracy, precision, recall, F1 score. Accuracy is the number of correctly predicted data points out of all the data points. A_{SLR} can be calculated as the number of all correct predictions to the total number of items in the data measures, shown in equation (3).

$$A_{SLR} = \frac{TP + TN}{TP + TN + FP + FN} \quad (3)$$

$$P_{SLR} = \frac{TP}{TP + FP} \quad (4)$$

$$R_{SLR} = \frac{TP}{TP + FN} \quad (5)$$

P_{SLR} describes how accurate our model is out of those predicted positives, how many of them are actual positive. P_{SLR} is a good measure to determine, when the cost of False positive is high. R_{SLR} calculates how many of the actual positives our model capture by labelling them as positive. R_{SLR} represent the model metric we will select when there is a high cost associated with False Negatives. The mathematical formulation of Precision and Recall are given in equation (4) and (5) respectively.

$$F_{SLR} = \frac{2 \times P \times R}{P + R} \quad (6)$$

F-Measure in equation (6) provides a way to combine both precision and recall into a single measure that captures both properties. It is used to handle imbalanced classification. Confusion matrix was also analyzed to have a better understanding of the types of errors being made by our classifier. The key to confusion matrix is number of correct and incorrect predictions are summarized with count values and broken down by each class.

5. Result and Discussion

A K-Fold Cross-Validation was performed on the dataset by taking ten folds. The average accuracy over ten iterations of different algorithms is demonstrated in Table 2. It can be observed from the presented accuracies that SVM outperformed other machine learning algorithms such as KNN, Random Forest, Decision Tree, Naïve Bayes and also achieved higher accuracy than deep learning algorithms such as Artificial Neural Network (ANN) and Multi-Layer Perceptron (MLP).

Table 2: Average accuracy obtained using machine learning and deep learning algorithms.

Dataset	SVM	KNN	Random Forest	Decision Tree	Naive Bayes	ANN	MLP
ASL(alphabet)	99.15%	98.21%	98.57%	98.57%	53.74%	97.12%	94.69%
Indian(alphabet)	99.29%	98.87%	98.59%	98.59%	86.77%	94.79%	96.48%
Italian(alphabet)	98.19%	96.75%	97.83%	97.83%	77.19%	78.63%	72.14%
ASL(numbers)	99.18%	99.18%	97.56%	97.56%	96.74%	95.12%	97.56%
Turkey (numbers)	96.22%	93.08%	94.33%	94.33%	83.64%	93.71%	83.64%

The highest accuracy achieved using the model is bolded in the above table for each of the sign language datasets.

For exhaustive testing, each sign language image dataset is pre-processed to extract features using MediaPipe framework and trained in Support Vector Machine to classify gestures correctly. An accuracy of 99% is achieved for most of the datasets which outperform present state-of-arts and classify fingerspellings of Sign Languages precisely. Maximum accuracy of 99.29% is gained for Indian Sign Language and minimum accuracy of 96.22% is obtained for Turkey Sign Language numbers prediction using handgestures. The testing performance for each dataset is summarized in Table 3. Confusion matrix is illustrated in figure 3 and figure 4 demonstrates real-time sign language detection.

Table 3: Performance analysis using SVM algorithm on different datasets

Dataset name	Training Accuracy	Testing Accuracy	Precision	Recall	F1-Score
ASL(alphabet)	99.50%	99.15%	99.15%	99.15%	99.15%
Indian(alphabet)	99.92%	99.29%	99.29%	99.29%	99.29%
Italian(alphabet)	99.72%	98.19%	98.19%	98.19%	98.19%

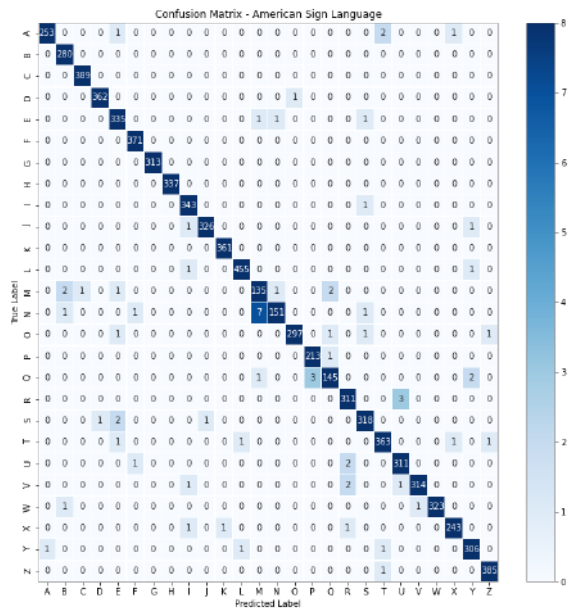
Turkey (numbers)	99.37%	96.22%	96.22%	96.22%	96.22%
American (numbers)	98.77%	99.18%	99.18%	99.18%	99.18%

The trained model is explicitly lightweight which makes our machine learning model appropriate for deployment in mobile application. Real-time sign language detection makes our methodology fast, robust, adaptable specifically for smart devices. Mediapipe's state-of-art makes feature extraction easy by breaking down and analyzing complex hand-tracking information, without the need to build a convolutional neural network from scratch. The proposed methodology uses minimum computational power and consumes less time to train model than other state-of-arts present. Table 4 illustrates comparison of the performance of other works of literature using machine learning / deep learning algorithms and ours.

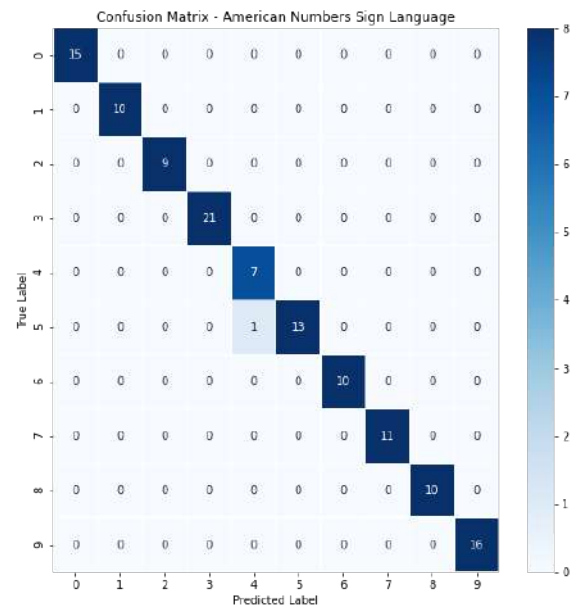
Table 4: Comparison with other current methods.

Sign Language	Reference	Type	Number of classes	Method	Accuracy
American	P.Das et al.,[14]	Alphabets	26	Deep CNN	94.3%
	M.Taskiran et al.,[15]	Alphabets and Numbers	36	CNN	98.05%
	N.Saquib and A.Rahman[16]	Alphabets	24	KNN	96.14%
				Random Forest	96.13%
				ANN	95.87%
				SVM	94.91%
	Ours	Alphabets	26	SVM	99.15%
		Numbers	10	SVM	99.18%
Indian	K.K.Dutta et al.,[17]	Alphabets	24	KNN	94%-96%
	M.Sharma et al.,[18]	Numbers	10	KNN and Neural Network	97.10%
	J.L.Raheja et al.,[19]	Alphabets	24	SVM	97.5%
	Ours	Alphabets	26	SVM	99.29%
Italian	L.Pigou et al.,[20]	Alphabets	20	CNN	91.7%
	Ours	Alphabets	22	SVM	98.19%

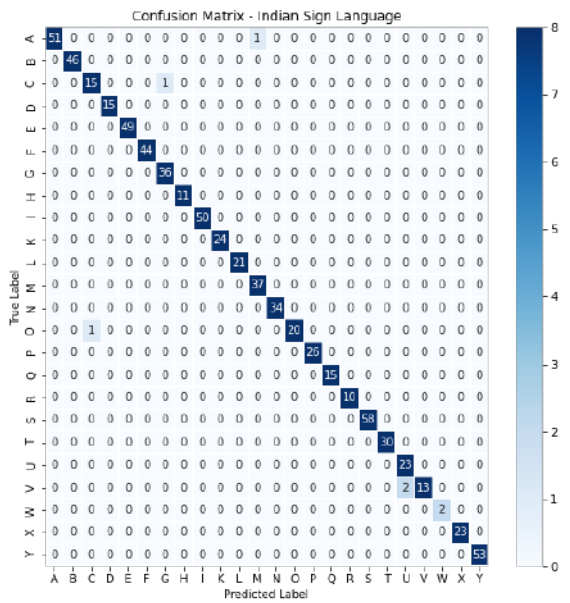
The highest accuracy is bolded in the above table for each of the sign language dataset.



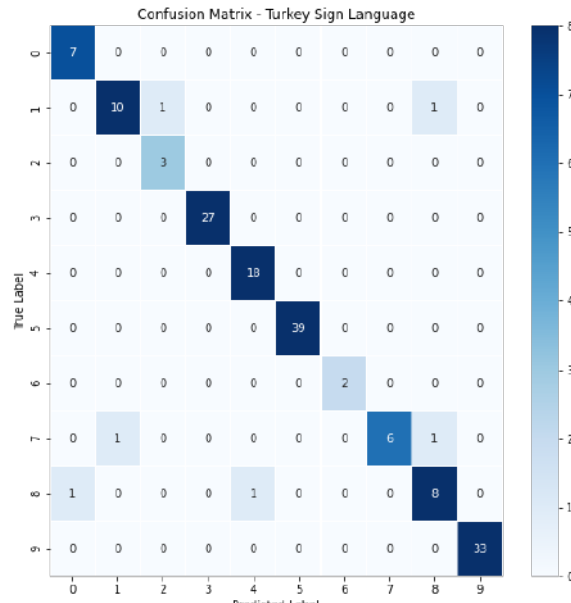
(a)



(b)



(c)



(d)

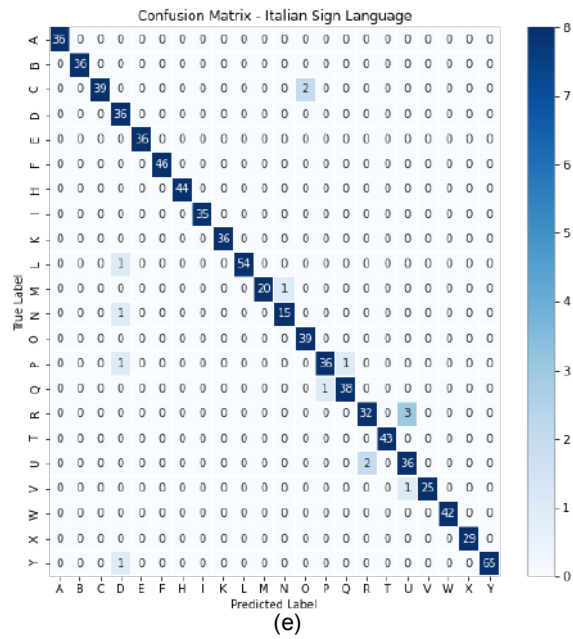


Figure 3: Confusion matrix a) American Sign Language (alphabets), b) American Sign Language (numbers), c) Indian Sign Language (alphabets), d) Turkey Sign Language (numbers), e) Italian Sign Language (alphabets)

American Sign Language - Alphabets



American Sign Language - Numbers



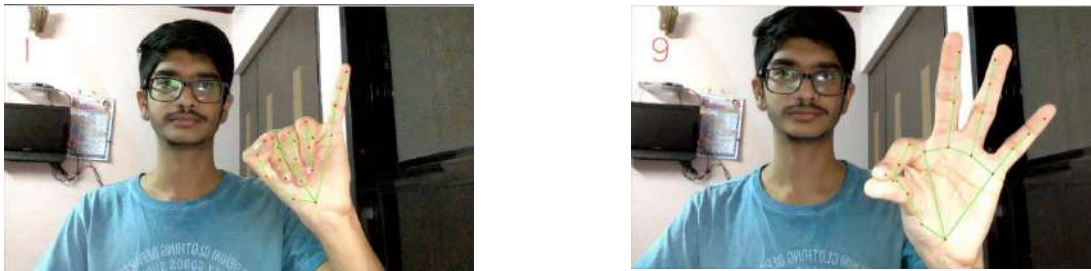


Figure 4: Real-time American Sign Language Recognition. American alphabets: 'S', 'U', 'I' and numbers: '1', '3', '9'

6. Conclusion

With an average accuracy of 99% in most of the sign language dataset using MediaPipe's technology and machine learning, our proposed methodology show that MediaPipe can be efficiently used as a tool to detect complex hand gesture precisely. Although, sign language modelling using image processing techniques has evolved over the past few years but methods are complex with a requirement of high computational power. Time consumption to train a model is also high. From that perspective, this work provides new insights into this problem. Less computing power and the adaptability to smart devices makes the model robust and cost-effective. Training and testing with various sign language datasets show this framework can be adapted effectively for any regional sign language dataset and maximum accuracy can be obtained. Faster real-time detection demonstrates the model's efficiency better than the present state-of-arts. In the future, the work can be extended by introducing word detection of sign language from videos using Mediapipe's state-of-art and best possible classification algorithms.

REFERENCES

- [1] Shukor AZ, Miskon MF, Jamaluddin MH, Bin Ali F, Asyraf MF, Bin Bahar MB. 2015. A new data glove approach for Malaysian sign language detection. *Procedia Comput Sci* 76:60–67
- [2] Almeida SG, Guimarães FG, Ramirez JA. 2014. Feature extraction in Brazilian sign language recognition based on phonological structure and using RGB-D sensors. *Expert Syst Appl* 41(16):7259–7271
- [3] Murakami K, Taguchi H. 1991. Gesture recognition using recurrent neural networks. In: *Proceedings of the ACM SIGCHI conference on Human factors in computing systems*, pp 237–242. <https://dl.acm.org/doi/pdf/10.1145/108844.108900>
- [4] Wang RY, Popović J. 2009. Real-time hand-tracking with a color glove. *ACM Trans Graph* 28(3):63
- [5] Rekha J, Bhattacharya J, Majumder S. 2011. Hand gesture recognition for sign language: a new hybrid approach. In: *International Conference on Image Processing, Computer Vision, and Pattern Recognition (ICCV)*, pp 80–86
- [6] Kurdyumov R, Ho P, Ng J. 2011. Sign language classification using webcam images, pp 1–4. <http://cs229.stanford.edu/proj2011/KurdyumovHONG-SignLanguageClassificationUsingWebcamImages.pdf>
- [7] Tharwat A, Gaber T, Hassanien AE, Shahin MK, Refaat B. 2015. Sift-based arabic sign language recognition system. In: *Springer Afro-European conference for industrial advancement*, pp 359–370. https://doi.org/10.1007/978-3-319-13572-4_30
- [8] Baranwal N, Nandi GC. 2017. An efficient gesture based humanoid learning using wavelet descriptor and MFCC techniques. *Int J Mach Learn Cybern* 8(4):1369–1388
- [9] Elakkiya R, Selvamani K, Velumadhava Rao R, Kannan A. 2012. Fuzzy hand gesture recognition based human computer interface intelligent system. *UACEE Int J Adv Comput Netw Secur* 2(1):29–33 (ISSN 2250–3757)
- [10] Ahmed AA, Aly S. 2014. Appearance-based arabic sign language recognition using hidden markov models. In: *IEEE International Conference on Engineering and Technology (ICET)*, pp 1–6. <https://doi.org/10.1109/ICETech.2014.7016804>
- [11] R. Sharma, R. Khapra, N. Dahiya. June 2020. Sign Language Gesture Recognition, pp.14-19
- [12] W. Liu, Y. Fan, Z. Li, Z. Zhang. Jan 2015. Rgb-d video based human hand trajectory tracking and gesture recognition system in Mathematical Problems in Engineering.
- [13] Zhang, F., Bazarevsky, V., Vakunov, A., Tkachenka, A., Sung, G., Chang, C. L., & Grundmann, M. 2020. MediaPipe Hands: On-device Real-time Hand Tracking. *arXiv preprint arXiv:2006.10214*.
- [14] Das, P., Ahmed, T., & Ali, M. F. 2020, June. Static Hand Gesture Recognition for American Sign Language using Deep Convolutional Neural Network. In *2020 IEEE Region 10 Symposium (TENSYP)* (pp. 1762-1765). IEEE.
- [15] M. Taskiran, M. Killioglu and N. Kahraman. 2018. A Real-Time System for Recognition of American Sign Language by using Deep Learning, 2018 41st International Conference on Telecommunications and Signal Processing (TSP), Athens, Greece, pp. 1-5, doi: 10.1109/TSP.2018.8441304.
- [16] Nazmus Saquib and Ashkur Rahman. 2020. Application of machine learning techniques for real-time sign language detection using wearable sensors. In *Proceedings of the 11th ACM Multimedia Systems Conference (MMSys '20)*. Association for Computing Machinery, New York, NY, USA, 178–189. DOI:<https://doi.org/10.1145/3339825.3391869>
- [17] Dutta, K. K., & Bellary, S. A. S. 2017, September. Machine learning techniques for Indian sign language recognition. In *2017 International Conference on Current Trends in Computer, Electrical, Electronics and Communication (CTCEEC)* (pp. 333-336). IEEE.
- [18] Sahoo, Ashok. 2014. Indian sign language recognition using neural networks and kNN classifiers. *Journal of Engineering and Applied Sciences*. 9. 1255-1259.
- [19] Raheja JL, Mishra A, Chaudary A. 2016 September. Indian Sign Language Recognition Using SVM I. *Pattern Recognition and Image Analysis*.; 26(2).
- [20] Pigou, L., Dieleman, S., Kindermans, P. J., & Schrauwen, B. 2014, September. Sign language recognition using convolutional neural networks. In *European Conference on Computer Vision* (pp. 572-578). Springer, Cham.