VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

CSE3999 - Technical Answers for Real World Problems

Indian Sign Language Recognition Using Distributed Machine Learning

Review - 3

Summer Semester Special 2021 – 2022

Slot - C1

Shriniket Dixit : 19BCE0389
Pratham Gupta : 19BCE0448
Jinay Bafna : 19BCI0117

DECLARATION

I hereby declare that the project entitled “Indian Sign Language Recognition Using Distributed Machine Learning" submitted by me, for the award of the degree of Bachelor of Technology in Programme to VIT is a record of bonafide work carried out by me under the supervision of Jothi K R. I further declare that the work reported in this thesis has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Place : Vellore

Date : 25 – 08-2022

CERTIFICATE

This is to certify that the thesis entitled “Indian Sign Language Recognition Using Distributed Machine Learning” submitted by Shriniket Dixit (19BCE0389), Pratham Gupta(19BCE0448), Jinay Bafna(19BCI0117) VIT University, for the award of the degree of Bachelor of Technology in Programme, is a record of bonafide work carried out by him under my supervision during the period, 15. 07. 2022 to 25.08.2022, as per the VIT code of academic and research ethics. The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university. The thesis fulfills the requirements and regulations of the University and in my opinion meets the necessary standards for submission.

Place : Vellore

Date : 25-08-2022

ACKNOWLEDGEMENTS

I would like to express my special thanks of gratitude towards my instructor Dr. Jothi K R for the course Technical Answers for Real World Problems (CSE3999) for encouraging my group to take up this project and for guiding us throughout the making of this project by giving valuable suggestion to improve the project. I would also like to thank my group members for their proactive role in making this project

Place: Vellore

Date:25/08/2022

Index Page No.

Scope

In this project we aim to give an Indian Sign Language Detection Model using distributed machine learning model. The distributed machine learning model achieved a significant speed up when compared to a sequential model. This opens up the possibility of utilizing distributed machine learning model in real Life applications where complex and large machine learning models are required. The use of distributed machine learning will help in speeding up the training process in these cases. In future, complexity of the neural network could be increased and a more enriched dataset with variations in background and light could be used to mimic real life scenarios to get further insights into the performance of distributed machine learning techniques

Abstract

Communication is one of the major areas where differently abled (deaf and dumb) people face problems on everyday basis. Recent advances in technology have helped in development of Sign Language Recognition systems and models which has helped in solving this problem to a great extent. Machine Learning based Sign Language Recognition models have shown to have high accuracy.

However due to large volume of data required to train the model, the training can be considerably long. To counter this problem, we propose an accelerated Indian Sign Language Recognition Model using Distributed Learning. The model uses Convolutional Neural Network(CNN) for recognizing signs and converting them to English language. Tensor Processing Units(TPUs)and Graphical Processing Units(GPUs)have been utilized for distributing the model training process.

Introduction

Sign Language is used to communicate through visual medium by using signs, gestures and facial expression. Deaf and dumb people utilize Sign Language to communicate with others. Different regions and countries have their own versions of Sign Languages. Indian Sign Language (ISL)is used in India and some neighboring countries. However, Sign Language is not understood by most people with normal hearing abilities. This creates a huge communication gap which can lead to lot of misunderstanding.

To solve this problem many Sign Language Recognition models which can detect the hand gestures have been developed. Deep Learning methods inspired by the working of biological neural networks have provided significant breakthroughs in the field of Sign Language Recognition. Recent studies have shown that neural network-based models can have accuracy rate of more than 90%.

Literature Survey

In recent years many approaches for sign language recognition using different technologies and methodologies have been proposed using electronic devices such as smart gloves or using machine learning technologies. Some researchers have utilized traditional machine learning methods while other have used deep learning methods for development of models for recognition of Sign Language. Various Sign Language datasets of different Sign Languages have been used by the researchers.

Following are some of the works done in the field of sign language recognition.

Indian Sign Language Recognition Using Machine Learning Techniques

This study describes a method for automatic recognition of ISL immobile numeric signs, where the signs were solely acquired using a regular digital camera, and electrical signals were captured without the usage of wearable equipment. Since the technology is designed to translate isolated digit signs into text, each sign image entered must contain exactly one numerical sign. A sign database with ISL digits has been constructed that has 5000 photos, 500 images for each numerical sign, in order to recognise ISL sign images in a real-time context (0-9). In terms of classification accuracy, the k-Nearest Neighbour classifier performs better than the Naive Bayes classifier.

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

There is no denying that the deaf-mute community has communication challenges in daily life. Artificial intelligence advancements in recent years have broken down this communication barrier. This paper's primary goal is to present an approach for automating Sign Language Recognition utilising the open-source MediaPipe framework and machine learning algorithm. The prediction model is portable and compatible with modern technology. Various sign language datasets from the US, India, Italy, and Turkey are utilised for training and testing the framework's functionality. The suggested model is effective, precise, and robust with an average accuracy of 98%. Usage of this technology is made more convenient and pleasant by real-time precise identification utilising the Support Vector Machine (SVM) algorithm without the use of wearable sensors.

Sign Language Transformers: Joint End-to-end Sign Language Recognition and Translation

Recognizing and comprehending sign language is a crucial component of the sign language translation endeavour. The initial stage of previous translation methods' systems mainly relied on recognition. In this research, we introduced Sign Language Transformers, an unique transformer-based architecture to learn sign language translation and recognition simultaneously. Instead of having an explicit gloss representation act as a data bottleneck, it used CTC loss to inject gloss level

supervision into the transformer encoder and train it to recognise signs while learning meaningful representations for the purpose of translating signs.

Real-Time Sign Language Detection using Human Pose Estimation

It recognises the necessity for such a case in videoconferencing and suggests a simple real-time sign language identification technique. Using a linear classifier and human pose estimate, it extracts optical flow features. It displays increases of up to 91 percent accuracy when a recurrent model is applied directly to the input, and it still operates in less than 4ms. Estimation is carried out in order to carry out a binary classification for each frame: is the signer present or not. It contrasts the acquisition times of various potential inputs, including bounding boxes, partial pose estimation, and full-body pose estimation. The method is tested on the Public DGS Corpus (German Sign Language), with results ranging from 87 to 91 percent prediction accuracy, depending on the input, with an inference time of 350 to 3500 s per frame.

A depth-based Indian Sign Language recognition using Microsoft Kinect

Recognition of sign language by a system has become important to bridge the communication gap between the abled and the Hearing and Speech Impaired people. This paper introduces an efficient algorithm for translating the input hand gesture in Indian Sign Language (ISL) into meaningful English text and speech. The system captures hand gestures through Microsoft Kinect (preferred as the system performance is unaffected by the surrounding light conditions and object colour). The dataset used consists of depth and RGB images with 140 unique gestures of the ISL taken from 21 subjects, which includes singlehanded signs, double-handed signs and fingerspelling totalling to 4600 images. They had managed to improve the average recognition accuracy up to 71.85% with the new method. They achieved 100% accuracy for the signs representing 9, A, F, G, H, N and P. However, their method does not take into account the context of gestures, leading to erroneous translations on a few occasions.

Indian Sign Language Recognition Using SVM

Recognition of sign language is a crucial field where individuals would benefit greatly from easy interaction with humans or machines. 2.8 million persons in India currently have speech or hearing impairments. Based on dynamic hand gesture recognition algorithms in a real-time scenario, this paper focuses on the recognition of Indian signs. For pre-processing, the recorded video was transformed to HSV colour space, and then segmentation based on skin pixels was carried out. Additionally, depth data was employed concurrently to produce more accurate results. From the picture frames, moments and motion trajectories were recovered, and Support Vector Machine was used to classify the gestures. Both a webcam and MS Kinect were used to evaluate the system.

A system like this would be useful for teaching and communicating with those who have hearing loss. The proposed system has 30 frames set. The suggested device is intended to help disabled people who cannot hear and have difficulty communicating through sign language. Four gestures were categorised using SVM as part of the implementation of the Indian sign language recognition system. The accuracy of the results was 97.5 percent, which was promising. To create a

comprehensive and reliable system for mobile platforms, researchers will expand on this work to cover all indications. Other regional languages may also be implanted using this method.

Real Time Indian Sign Language Recognition System to aid Deaf-dumb People

People who are deaf or dumb can communicate through sign language. In this research, a technique was put forth that can be used to create a Sign Language Recognition system for one of the south Indian languages. A set of 32 signs that each represent the binary "UP" and "DOWN" positions of the five fingers are defined in the suggested method. The photos, which are of the right hand's palm side, are dynamically loaded at runtime. During the method's development and testing phases, a single user was taken into consideration. The pre-processed static photos were created using the feature point extraction approach, and each sign's training set consists of 10 images. By utilising image processing algorithms to analyse static photos, the images are transformed into text by locating the fingertip position. The suggested method can recognise the signer's photos that are dynamically taken during testing. The test image findings demonstrate that, after training with 320 photos and testing with 160 images, the proposed Sign Language Recognition System can identify images with an accuracy of 98.125 percent. Using image processing techniques, a suggested sign language recognition system for human-computer interaction was developed effectively with accuracy on par with more recent contributions. The results demonstrate that the signals with a lower recognition rate can be improved by employing the suggested angular measurement. The 32 signs that have been created can be used to identify the letters of Tamil, one of the south Indian languages with 12 vowels and 18 consonants. This opens the door for later work.

Deep Learning Methods for Indian Sign Language Recognition

In this paper a method to implement real time Indian Sign Language gesture recognition using two methods. First using a depth+RGB based Microsoft Kinect camera and then using a normal RGB camera. For depth+RGB based techniques, the hand segmentation was done using depth perception techniques. For a normal RGB camera semantic segmentation approach was adopted. The usage of semantic segmentation completely removes the necessity of using a depth-based camera and segmentation. For the depth+RGB trained models, techniques like having different lighting conditions and data augmentation helped achieve the generalization in case of static gestures. For dynamic gestures, the procurement of data was done at various fps values so that the model can learn the temporal features.

Artificial Neural Network based Indian Sign Language Recognition using hand crafted features

This article presents a methodology to recognize Indian Sign Language (SL) gestures and translate them into English. SL Recognition systems can be useful for facilitating the conversation. There are various systems developed by researchers for implementing a SL recognition system. Being in its developing stage, the grammar rules of Indian Sign Language (ISL) are not documented making the recognition process a challenge. This approach employs hand crafted feature extraction technique

and uses Artificial Neural Network for classification of the gestures. The accuracy of model achieved is as high as 98% using this methodology.

Recognition of Sign Language Using Leap Motion Controller Data

The paper reviews the state of the art in sign language recognition. The study focuses on data acquisition methods, because they put the major restrictions on a further recognition process. The sign language recognition, using data collected by the Leap Motion Controller, is performed. The device detects user's hands and transmits it into a three-dimensional model, providing hands and lingers location and posture. This paper presents a system that is capable of recognition and classification of the sign language using deep convolutional neural network. It highlights the main features of the Leap Motion Controller that improves and simplifies feature extraction.

Deep Learning-Based Approach for Sign Language Gesture Recognition with Efficient Hand Gesture Representation

This study proposed a novel system for dynamic hand gesture recognition via a combination of multiple deep learning techniques. The proposed system represented the hand gesture using local hand shape features as well as global body configuration features, which is very efficient for complicated structured hand gestures of the sign language. The openpose framework was used in this study for hand region detection and estimation. A robust face detection algorithm and the body parts ratios theory were utilized for gesture space estimation and normalization. Two 3DCNN instances were used separately for learning the fine-grained features of the hand shape and the coarse-grained features of the global body configuration. MLP and autoencoders were utilized to aggregate and globalize the extracted local features and the SoftMax function was used for the classification. The experimental results showed that the proposed system outperformed state-of-the-art methods in terms of recognition rate, demonstrating its effectiveness.

Signet: A Deep Learning based Indian Sign Language Recognition System

The paper presented a vision based deep learning architecture for signer independent Indian sign language recognition system. Here, various existing methods in sign language recognition and implement a Convolutional Neural Network (CNN) architecture for ISL static alphabet recognition from the binary silhouette of signer hand region were reviewed. The system was successfully trained on all 24 ISL static alphabets with a training accuracy of 99.93% and with testing and validation accuracy of 98.64%. The recognition accuracy obtained is better than most of the current state of art methods.

Motionlets Matching with Adaptive Kernels for 3-D Indian Sign Language Recognition

This paper proposes characterization of sign language gestures articulated at different body parts as 3-D motionlets, which describe the signs with a subset of joint motions. A two-phase fast algorithm identifies 3-D query signs from an adaptively ranked database of 3-D sign language. A model for recognizing gestures of Indian sign language 3D motion captured data is presented. It is observed that the motionlet based adaptive kernel matching algorithm on 500 class 3D sign language data gives better classification accuracies compared to state-of-the-art action recognition models.

Evaluation of CNN Models with Transfer Learning for Recognition of Sign Language: Alphabets with Complex Background

A Convolutional Neural Network (CNN) approach with transfer learning to recognize Arabic and American Sign Languages alphabets with complex background is presented. Different techniques to improve the accuracy of the proposed approach such as data augmentation, batch-normalization, and early stopping were adopted. The proposed model is evaluated on three datasets and experiments reveal improved results with high recognition rates. They could have explored more about hand segmentation techniques as it is a very important method for developing a sign language.

A Static Hand Gesture Based Sign Language Recognition System using Convolutional Neural Networks

In this paper, a SL interpreter that takes the input sign gesture and gives the output in a display device is developed. They used Convolutional Neural Networks (CNNs) to train the system with a given database. After training we find out the testing accuracy of 99.89% and validation accuracy of 99.85% at 5 epoch. One of the advantages of our model is that it is not dependent on external hardware or device The limitation is that the background must be light with good lighting conditions. Also it is applicable only for static gestures

Sign Language Recognition Techniques- A Review

In this paper it is aimed to review various techniques that have been employed in the recent past for SLR that are employed at various stages of recognition. This paper explains the methods of Sign language recognition and describes the steps involved in gesture recognition which include acquisition, segmentation, feature extraction till recognition and classification. But this paper in itself didn't discuss or produce a new method to improve the topic.

FineHand: Learning Hand Shapes for American Sign Language Recognition

In this paper, they present an approach for effective learning of hand shape embedding, which are discriminative for ASL gestures. They demonstrated that higher quality hand shape models can significantly improve the accuracy of final video gesture classification in challenging conditions with variety of speakers, different illumination and significant motion blur.

Alphabetical Gesture Recognition of American Sign Language using E-Voice Smart Glove

In this paper the authors have given the idea of a smart glove that has been designed to be an interpreter between deaf-mute individuals and the normal public. Gesture translation into speech and textural form is carried out through this device. The smart glove is embedded with modern and technologically advanced sensors to make the overall prototype lightweight and easy to carry. Gesture translation is performed by idealizing the standard ASL template. The main problem encountered in this paper is that the idea is not implementable on large scale as the person might not have the means of buying these gloves.

Sign Language Recognition with CW Radar and Machine Learning

In another work, the authors have explored the use of low power frequency modulated continuous wave radar for automatic sign language recognition. The proposed system is composed of a radar, a sound cluster, and a computer for transforming signals to spectrograms. Furthermore, as the time-frequency spectrograms are high-dimensional data with redundant information, then dimensionality reduction is performed by extracting the histogram of oriented gradients features from these pectrograms. The features are finally classified by the k-Nearest Neighbor algorithm and a classification result of 95.8% is achieved on the five testing signs. The impact of the k value in the k-Nearest Neighbor is also investigated in this research paper. However, the proposed method is not easy to implement and users can find it hard to use it for daily conversation.

Wearable Sensor-Based Sign Language Recognition: A Comprehensive Review

This is a review paper where the authors have prepared a literature review focuses on analyzing studies that use wearable sensor-based systems to classify sign language gestures. A review of 72 studies from 1991 to 2019 was performed to identify trends, best practices, and common challenges. Attributes including sign language variation, sensor configuration, classification method, study design, and performance metrics were analyzed and compared. Many encouraging methods and results related to this field were observed, and common challenges were identified and analyzed. Major challenges of SLR include sign boundary detection, system scalability to larger lexicons, eliminating movement epenthesis, and model convergence. Although attempts have been made to overcome these challenges, techniques are still being developed by researchers.

ST-Xception: A Depthwise Separable Convolution Network for Military Sign Language Recognition

In this paper, the authors have collected a new first-person dataset named MSL which contains 16 classes of 3, 840 tactical gesture samples on battle scenario with more than 11, 0000 video frames performed by 10 subjects. Moreover, they have also presented a novel deep network, called ST-Xception architecture, considering the depth wise separable convolutions to recognize such military sign language. By expanding the convolution filters and pooling kernels into 3D, our network can characterize the inherent spatio-temporal relationship of a certain tactical hand gesture. In particular, they have also further reduced computational cost and relieve overfitting by replacing the fully connected layers with adaptive average pooling Experimental results show that their model outperforms existing models on their in house MSL dataset and as well as in two other benchmark datasets. The approach (expansion of convolution filters and pooling of kernels in 3D) applied in this research paper is a very novel one and can considered for further analysis.

A Brief Review of the Recent Trends in Sign Language Recognition

In this review paper the author has tried to review both the most commonly used method of sign language recognition. The author talks about two main approaches for sign language recognition that are image based and sensor based. Image based approach involves one or more cameras to capture an image sequence of the signer performing the sign, and then uses image processing to recognize the sign. The sensor-based method uses instrumental gloves assembled with sensors to track the hand articulates. This paper mainly describes various image or vision-based sign language recognition systems comprising feature extraction and classification. Translation of sign language to speech is also described briefly Overall, this paper is expected to be a complete introduction to automatic hand gesture recognition and sign language interpretation.

Transfer Learning for Videos: From Action Recognition to Sign Language Recognition

In this work researchers have proposed have proposed model for signer independent sign language recognition Inflated 3D Convolutional Neural Network has been employed for sign language recognition. The presented method utilizes solely RGB video data. The proposed model can also work on application which do not provide or have access to depth data. The researchers have also demonstrated that transfer of spatiotemporal features to training process for SLR from a large scale action recognition dataset can be highly beneficial. The presented work was evaluated using the ChaLearn249 Isolated Gesture Recognition dataset and has accuracy of 64.44%. The proposed model performed significantly better than many other recent RGB-based methods.

Real-Time Sign Language Recognition Based on Video Stream

A model for real time Chinese Sign Language Recognition has been proposed. The researchers created a dataset for the Chinese Language containing nearly 5000 words along with their demos. RGB video streams are used as the input for model. Optical Flow Calculation has been used for preprocessing the pixels. After preprocessing, the video stream was given as input to 30 Convolutional Neural Network which has the ability to extract both time and space features for extraction of feature vectors. To increase the practicality of the system, motion detection module, hand and head detection module along with an artificial interaction Interface was embedded into the system. The proposed model showed highest average accuracy of 90.1% when 3D CNN was used along with Optical flow processing with RGB video stream

Towards Multilingual Sign Language Recognition

Research has also been done to develop models for multilingual sign language recognition. In one such approach hand movement modelling was done with usage of target sign language independent dataset by derivation of subunits of hand movement. The proposed approach was validated against different types of Sign Language. This work demonstrated that sign language recognition models could be developed by utilizing multilingual sign language data. Although considerable performance difference has been observed when hand modelling is done in a language independent manner rather than in language dependent manner.

Recognition of Sign Language Symbols using Templates

This work proposes a model for the detection of Sign Language Symbols using facial expression and hand detection. The model uses webcam for image input and works only with static images. The input static images are taken in YCbCr format and skin color detection is performed for development of templates. The developed templates are divided into Quadrants and computation of quad values is carried out These computed quad values provide threshold values for the purpose of matching and recognition of Sign Language Symbols. The model employs image processing techniques such as feature detection for creation of a template image that can be matched against incoming input and recognize the Sign Language Symbol.

Reconstruction of Convolutional Neural Network for Sign Language Recognition

In this approach, Sign Language Recognition model using Convolutional Neural Network (CNN) has been proposed. American Sign Language fingerspelling dataset has been used in the model. Single Shot Multi-Box Detector (SSD) a machine learning model which is highly efficient and precise in object detection has been used for hand capturing and provide input to the Convolutional Neural Network (CNN). The CNN in combination with a fully connected network forms the second module of the model. The second module translates the input signs into text. The proposed model had an accuracy of 92.21% which was better than many of the existing works that utilized machine learning techniques for sign language recognition.

INCLUDE: A Large-Scale Dataset for Indian Sign Language Recognition

ISL is a fully developed language with its own grammar, syntax, lexicon, and other distinctive linguistic features. In India, more than 5 million deaf individuals utilise it. There isn't currently a publicly accessible dataset on ISL that can be used to test Sign Language Recognition (SLR) methods. They introduced the Indian Lexicon Sign Language Dataset - INCLUDE - in their paper, which is an ISL dataset made up of 0.27 million frames from 4,287 movies and 263 signs from 15 different word categories. To offer a close match to the environment, INCLUDE is recorded with the assistance of skilled signers. By fine-tuning the decoder for a dataset of American Sign Language, we further investigate generalisation. The model's accuracy on the ASLLVD with 48 classes is 92.1 percent, which is an improvement above previous findings and a useful way to assist SLR for many languages.

An integrated mediapipe-optimized GRU model for Indian sign language recognition

A language with distinct and intricate linguistic rules, sign language is a vision-based interactive language. People who have trouble hearing use other body parts to express and convey their emotions, ideas, and thoughts. Since sign language has a distinct linguistic framework, it varies geographically from one region to another. Each nation has created its own sign language for use by its deaf and hard-of-hearing populations. American Sign Language (ASL) in the US, British Sign Language (BSL) in the UK, Indian Sign Language (ISL) in India, and Korean Sign Language (KSL) in Korea are a few of the more well-known sign languages.

In this study, a MOPGRU model for ISL recognition was put out. They specifically changed the typical GRU cell's update gate by increasing its output by the reset gate. The output of the reset gate re-screens the information and removes the undesired information in the data, so drawing greater attention to the relevant information thanks to their better update gate mechanism.

Selfie video based continuous Indian sign language recognition system

In this research, a brand-new technique was presented for applying sign language to mobile platforms in real time. Selfie-captured sign language video is processed using only a smart phone's computer capability. A sign language feature space is created using pre-filtering, segmentation, and feature extraction on video frames. On the sign feature space, iteratively trained and tested classifiers using Minimum Distance and Artificial Neural Networks are used. The power of the Sobel Edge Operator is increased by morphology and adaptive thresholding, resulting in nearly flawless segmentation of the hand and head sections that accounts for the minute vibrations of the selfie stick. With an average Word Matching Score (WMS) of about 85.58 percent for MDC and 90 percentfor ANN and a modest variance of 0.3 s in classification times, the suggested technique performs well. This innovative selfie sign language recognizer application will undoubtedly become popular inapp stores thanks to neural network classifiers with quick training algorithms.

METHODOLOGY

The proposed model involves distributed learning which has been implemented using GPUs and TPUs. The dataset has been pre-processed to achieve better performance in term of accuracy.

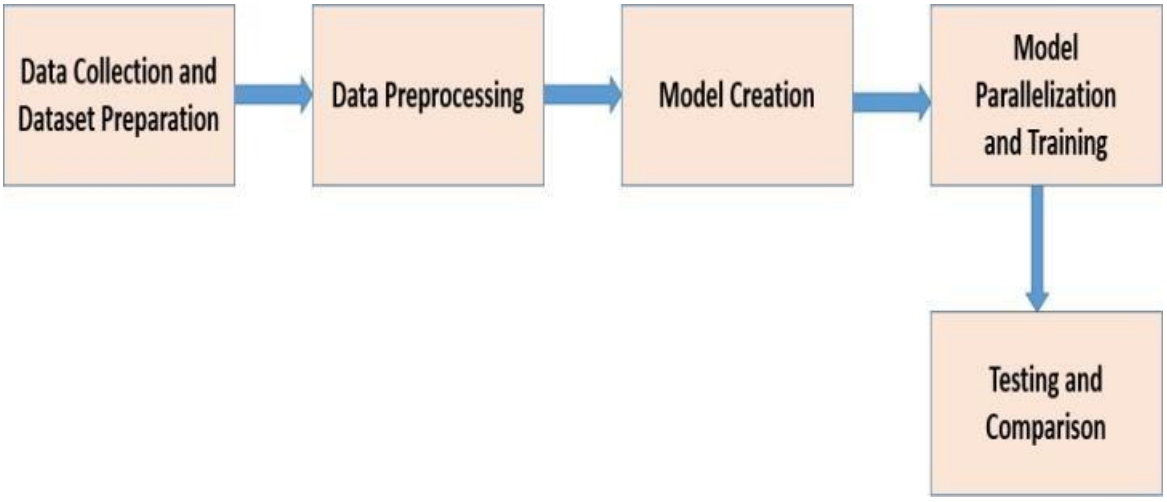


Fig.1.OverallArchitectureoftheProposedModel

Data Collection and Preparation

In the dataset we have 1000 images of each letter divided into the test and training portion. The size of the dataset is enough for proper training and testing of the model that we are trying to create. After this we have compressed our dataset into zip file for use in the Colab notebook.

Data Pre-processing

- The aim of the data preprocessing is to make dataset ready to be input into CNN for training and testing. Two major steps are involved in data preprocessing: Skin detection and segmentation followed by edge detection. OpenCV a well-known library for computer vision and image processing has been used for data preprocessing. For detection of hand gestures, it is necessary to identify the area of image representing hand and discard other unnecessary features. For skin detection, the unprocessed data (image) is converted to color space where Y represents the luma component while Cb and Cr represent blue and red difference chroma components respectively. This color space can be formed using RGB values through following equations.

- $Y=0.299R+0.587G+0.114BC$

- $Cr = R-Y$

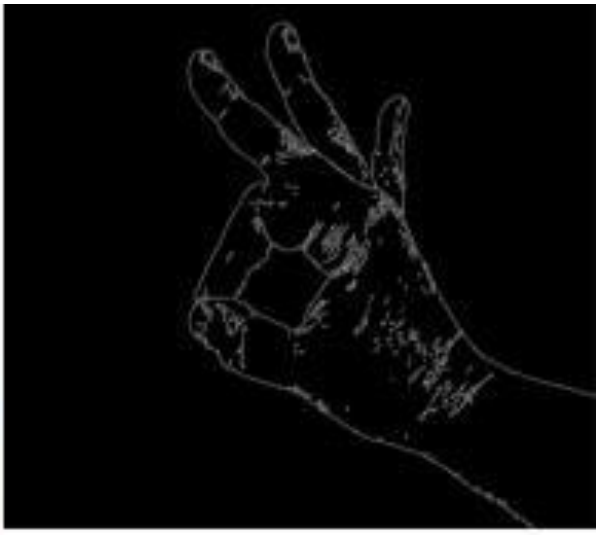
- $Cb = B - Y$

- After conversion to YCbCr color space, image histogram is formed to mark each skin-colored pixel. By this process only the skin-colored pixels representing hand remain all other unnecessary details such as background are removed.

- Hand Sign Image after Skin Detection has been performed

- After skin detection, it is necessary to identify the edges since all the hand signs are identified by the outline or edges of the hand. For this task, Canny Edge Algorithm has been utilized. Canny Edge Algorithm can not only has ability to detect a wide variety of edges but also can reduce noise that occurs during the process. Gaussian filter is used for the noise reduction followed by calculation of intensity gradient. Non- maximum suppression is carried out after gradient calculation, Hysteresis Thresholding is performed at end to remove any false edges.

- Canny edge detection Performed on image After Skin Detection



Model creation

- We created a Convolution Neural Network model here. In our neural network we first added a convolution layer which accepts the image input. After it one more convolution layer was added. A dropout layer was added as it prevents the overfitting.

- Again, the convolution layers were added and then dropout layer was added and then SoftMax layer was added which converts the number to vector probabilities. After this optimizer and loss function were added where we used the adam optimizer and categorical cross entropy for loss function. In final step metric was added as accuracy.

Model Parallelization using TPU

For model parallelization using TPU we first choose the hardware accelerator as TPU and then we connected the TPU by creating TPU Cluster Resolver which connects available TPU instances to our Colab notebook. After this we initialize our distribution strategy. Here resolver provide us access to TPU so that we create our distributed pipeline. After that we build our model under the distributed strategy which makes a normal model that runs on CPU to run in a TPU. In last step we simple train the model.

Model Parallelization using GPU

In this part first we initialized the GPU available in Google Colab notebook by using the hardware accelerator as GPU. Keras and Tensorflow packages are installed which support the GPU. After this when training the model Tensoflow by default will reserve all the memory of GPU for use.

Testing and Comparison

The performance of the TPU is a based distributed models are compared with sequential CPU based model. Following performance indicators have been used for the evaluating the performance of all models.

1) Training Time: This is the time taken by the model to train on the given dataset. It is calculated in seconds.

2) Accuracy. This parameter tells how accuracy the model recognizes the sign represented by the input image.

3) Speed Up: It is training time of the CPU divided by the Time taken by the Processor used.

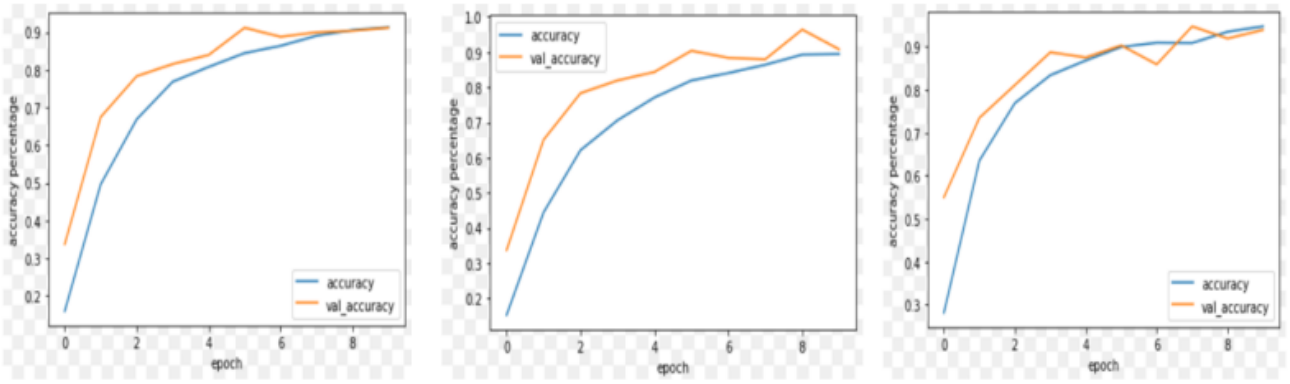
Result and Analysis

After testing of the sequential and distributed models, following results were obtained.

All three models achieved accuracy of greater than 98% which is adequate for any real life scenario.

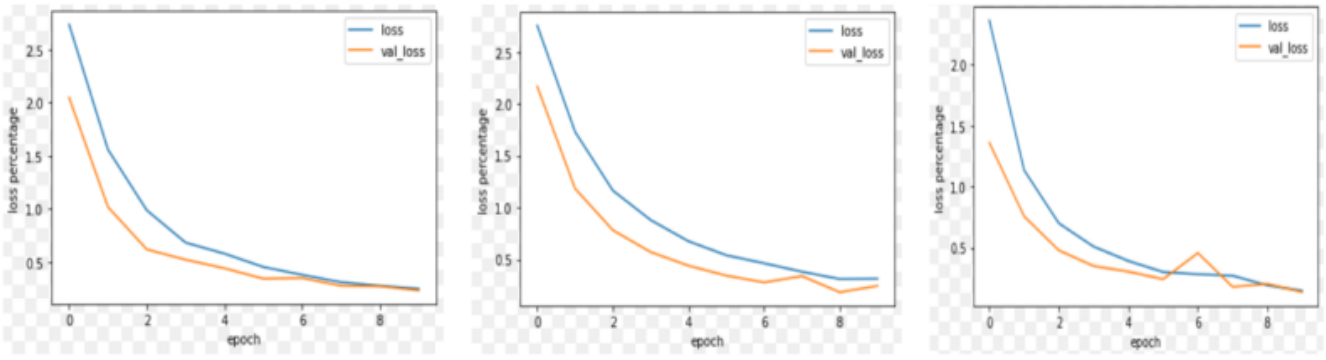
Result table showing Accuracy, Training Time and Speed Up of all processors.

Processor Used	Accuracy (%)	Training Time (seconds)	Speed Up
CPU	94.54	385	1
GPU	94.42	83	4.63
TPU	94.41	34	11.32



(a) (b) (c)

Fig.4. Accuracy Graph of (a) TPU Based Model (b) GPU Based Model (c) CPU Based Model



(a) (b) (c)

Fig.5. Training Loss Graph of (a) TPU Based Model (b) GPU Based Model (c) CPU Based Model

As results show distributed systems perform significantly better than CPU based system while GPU achieved a speed up of 4.63, TPU based model had a speed of 11.32 while maintaining a similar accuracy to a CPU based model. This shows us that TPU are highly

efficient and fast in distributed machine learning and can speed up the training process by a considerable margin outperforming CPU and GPU. This makes TPU adequate for development of machine learning models which require large datasets for training and testing.

Conclusion and Scope of Future Study

In this project we have given an Indian Sign Language Detection Model using distributed machine learning model. The distributed machine learning model achieved a significant speed up when compared to a sequential model. This opens up the possibility of utilizing distributed machine learning model in real life applications where complex and large machine learning models are required. The use of distributed machine learning will help in speeding up the training process in these cases. In future, complexity of the neural network could be increased and a more enriched dataset with variations in background and light could be used to mimic real life scenarios to get further insights into the performance of distributed machine learning techniques.

Code:

Data pre-processing: -

```
import glob
import cv2
import numpy as np
import os
dir = "Z"
p_dir = "C:/Users/dixit/Desktop/data"
pat=os.path.join(p_dir,dir)
os.mkdir(pat) cv_img = []
for img in glob.glob("C:/Users/dixit/Desktop/cancer/gettyimages
542502429 27867461d697a947417e7f8760b72b1c96880582-
s800c85.jpg"):
n = cv2.imread(img)
cv2.imshow('sin',n)
cv2.waitKey(0)
cv_img.append(n)
minRange = np.array([0, 133, 77], np.uint8)
maxRange = np.array([235, 173, 127], np.uint8)
YCRn = cv2.cvtColor(n, cv2.COLOR_BGR2YCR_CB)
skinArea = cv2.inRange(YCRn, minRange, maxRange)
detectedSkin = cv2.bitwise_and(n, n, mask=skinArea)
cv2.imshow('sin', detectedSkin)
cv2.waitKey(0)
edges = cv2dixit
edges = cv2.Canny(detectedSkin, 100, 200)
cv2.imshow('sin', edges)
cv2.waitKey(0)
cv2.imwrite("C:/Users/dixit/Desktop/data/Z/image%04i.jpg" %i,edges)
i+=1
cv2.destroyAllWindows()
```

- **Model training on CPU: -**

```
from google.colab import drive
drive.mount("/content/drive", force_remount=True)
!unzip "/content/drive/MyDrive/data.zip"

pip install tensorflow

import numpy as np
import pandas as pd#dataprocessing,CSVfileI/O(e.g.pd.read_csv)
import os
import cv2
#np.random.seed(5)
import tensorflow as tf
tf.random.set_seed(2)
import matplotlib.pyplot as plt
%matplotlib inline
from sklearn.model_selection import train_test_split
import keras
from keras.layers import Conv2D,MaxPool2D,Flatten,Dense,Dropout,BatchNormalization
from keras.preprocessing.image import ImageDataGenerator
from keras.models import Sequential
from keras import regularizers


train_dir='/content/data'
#test_dir='/content/data/isl_alphabet_test'

def load_unique():
  size_img=64,64
  images_for_plot=[]
  labels_for_plot=[]
  for folder in os.listdir(train_dir):
    for file in os.listdir(train_dir+'/'+folder):
      filepath=train_dir+'/'+folder+'/'+file
      image=cv2.imread(filepath)
      final_img=cv2.resize(image,size_img)
      final_img=cv2.cvtColor(final_img,cv2.COLOR_BGR2RGB)
      images_for_plot.append(final_img)
      labels_for_plot.append(folder)
      break
  return images_for_plot, labels_for_plot
```

```
images_for_plot,labels_for_plot=load_unique()
print("unique_labels=",labels_for_plot)
fig=plt.figure(figsize=(15,15))

def plot_images(fig,image,label,row,col,index):
 fig.add_subplot(row,col,index)
 plt.axis('off')
 plt.imshow(image)
 plt.title(label)
 return

#@title Default title text
image_index=0
row=5
col=6
for i in range(1,(row*col)):
 plot_images(fig,images_for_plot[image_index], labels_for_plot[image_index],row,col,i)
 image_index=image_index+1
plt.show()
labels_dict={'A':0,'B':1,'C':2,'D':3,'E':4,'F':5,'G':6,'H':7,'I':8,'J':9,'K':10,'L':11,'M':12,'N':13,'O'
:14,'P':15,'Q':16,'R':17,'S':18,'T':19,'U':20,'V':21,'W':22,'X':23,'Y':24,'Z':25}

from keras.utils import np_utils

def load_data():
 images=[]
 labels=[]
 size=64,64
 print("LOADING DATA FROM:",end="")

 for folder in os.listdir(train_dir):
  if(folder in '123456789'):
   continue
  print(folder,end='|')
  for image in os.listdir(train_dir+"/"+folder):
   temp_img=cv2.imread(train_dir+'/'+folder+'/'+image)
   temp_img=cv2.resize(temp_img,size)
   images.append(temp_img)
   labels.append(labels_dict[folder])
 images=np.array(images)
 images=images.astype('float32')/255.0
 labels=keras.utils.np_utils.to_categorical(labels)
 X_train,X_test,Y_train,Y_test=train_test_split(images,labels,test_size=0.05)
 print()
```

```
  print('Loaded',len(X_train),'images for training,','Train data shape=',X_train.shape)
  print('Loaded',len(X_test),'images for testing','Test data shape=',X_test.shape)
  return X_train,X_test,Y_train,Y_test

X_train,X_test,Y_train,Y_test=load_data()

from tensorflow.keras.utils import to_categorical
y_cat_train=to_categorical(Y_train,26)
y_cat_test=to_categorical(Y_test,26)
print(y_cat_train.shape)
print(y_cat_test.shape)

from keras.layers import Conv2D,MaxPooling2D
from keras.layers import Flatten,Dense
from keras.models import Sequential
from keras.layers import Activation

def create_model():
  model=Sequential()
  model.add(Conv2D(64,kernel_size=4,strides=1,activation='relu', input_shape=(64,64,3))
)
  model.add(Conv2D(64,kernel_size=4,strides=2,activation='relu'))
  model.add(Dropout(0.5))
  model.add(Conv2D(128,kernel_size=4,strides=1,activation='relu'))
  model.add(Conv2D(128,kernel_size=4,strides=2,activation='relu'))
  model.add(Dropout(0.5))
  model.add(Conv2D(256,kernel_size=4,strides=1,activation='relu'))
  model.add(Conv2D(256,kernel_size=4,strides=2,activation='relu'))
  model.add(Flatten())
  model.add(Dropout(0.5))
  model.add(Dense(512,activation='relu'))
  model.add(Dense(26,activation='softmax'))
  model.summary()
  return model

from tensorflow.keras.callbacks import EarlyStopping
early_stop=EarlyStopping(monitor='val_loss',patience=2)
model=create_model()
model.compile(optimizer='adam',loss='categorical_crossentropy',metrics=["accuracy"])
model_hist=model.fit(X_train,Y_train,epochs=5,batch_size=32,verbose=2)
score=model.evaluate(x=X_test,y=Y_test,verbose=0)
rint('Testaccuracy:',score[1])

prediction = model.predict(X_test[:1])
```

```
print("prediction shape:", prediction.shape)
X_test[:1]
```

- **Model training on TPU: -**

```
from google.colab import drive
drive.mount("/content/drive")
!unzip "/content/drive/MyDrive/data.zip"
pip install tensorflow

import tensorflow as tf
import numpy as np
import pandas as pd
import os
import keras
from keras.layers import Conv2D,MaxPool2D,Flatten,Dense,Dropout,BatchNormalization
from keras.preprocessing.image import ImageDataGenerator
from keras.models import Sequential
from keras import regularizers
from sklearn.model_selection import train_test_split
import cv2
import matplotlib.pyplot as plt
import seaborn as sns

try:
  tpu = tf.distribute.cluster_resolver.TPUClusterResolver() #TPUdetection
except ValueError:
  tpu=None
  print("None")

try:
  device_name=os.environ['COLAB_TPU_ADDR']
  TPU_ADDRESS='grpc://'+device_name
  print('FoundTPUat:{}'.format(TPU_ADDRESS))
except KeyError:
  print('TPUnotfound')

train_dir='/content/data'
#test_dir='/content/data/isl_alphabet_test'

def load_unique():
  size_img=64,64
  images_for_plot=[]
```

```
  labels_for_plot=[]
  for folder in os.listdir(train_dir):
    for file in os.listdir(train_dir+'/'+folder):
      filepath=train_dir+'/'+folder+'/'+file
      image=cv2.imread(filepath)
      final_img=cv2.resize(image,size_img)
      final_img=cv2.cvtColor(final_img,cv2.COLOR_BGR2RGB)
      images_for_plot.append(final_img)
      labels_for_plot.append(folder)
      break
  return images_for_plot,labels_for_plot

images_for_plot,labels_for_plot=load_unique()
print("unique_labels=",labels_for_plot)
fig=plt.figure(figsize=(15,15))

def plot_images(fig,image,label,row,col,index):
  fig.add_subplot(row,col,index)
  plt.axis('off')
  plt.imshow(image)
  plt.title(label)
  return

image_index=0
row=5
col=6
for i in range(1,(row*col)):
  plot_images(fig,images_for_plot[image_index],labels_for_plot[image_index],row,col,i)
  image_index=image_index+1
plt.show()
labels_dict={'A':0,'B':1,'C':2,'D':3,'E':4,'F':5,'G':6,'H':7,'I':8,'J':9,'K':10,'L':11,'M':12,'N':13,'O'
:14,'P':15,'Q':16,'R':17,'S':18,'T':19,'U':20,'V':21,'W':22,'X':23,'Y':24,'Z':25}

from keras.utils import np_utils

def load_data():
  images=[]
  labels=[]
  size=64,64
  print("LOADING DATA FROM:",end="")

  for folder in os.listdir(train_dir):
    if(folder in '123456789'):
      continue
```

```
    print(folder,end='|')
    for image in os.listdir(train_dir+"/"+folder):
      temp_img=cv2.imread(train_dir+'/'+folder+'/'+image)
      temp_img=cv2.resize(temp_img,size)
      images.append(temp_img)
      labels.append(labels_dict[folder])
  images=np.array(images)
  images=images.astype('float32')/255.0
  labels=keras.utils.np_utils.to_categorical(labels)
  X_train,X_test,Y_train,Y_test=train_test_split(images,labels,test_size=0.05)
  print()
  print('Loaded',len(X_train),'images for training,','Train data shape=',X_train.shape)
  print('Loaded',len(X_test),'images for testing','Test data shape=',X_test.shape)
  return X_train,X_test,Y_train,Y_test

X_train,X_test,Y_train,Y_test=load_data()
from tensorflow.keras.utils import to_categorical

y_cat_train=to_categorical(Y_train,26)
y_cat_test=to_categorical(Y_test,26)
print(y_cat_train.shape)
print(y_cat_test.shape)
resolver=tf.distribute.cluster_resolver.TPUClusterResolver(tpu='grpc://'+os.environ['COL
AB_TPU_ADDR'])
tf.config.experimental_connect_to_cluster(resolver)
tf.tpu.experimental.initialize_tpu_system(resolver)
print("Alldevices:",tf.config.list_logical_devices('TPU'))
print(len(tf.config.list_logical_devices('TPU')))
strategy=tf.distribute.TPUStrategy(resolver)

from keras.layers import Conv2D,MaxPooling2D
from keras.layers import Flatten,Dense
from keras.models import Sequential
from keras.layers import Activation

def create_model():
  model=Sequential()
  model.add(Conv2D(64,kernel_size=4,strides=1,activation='relu',
  input_shape=(64,64,3)))
  model.add(Conv2D(64,kernel_size=4,strides=2,activation='relu'))
  model.add(Dropout(0.5))
  model.add(Conv2D(128,kernel_size=4,strides=1,activation='relu'))
  model.add(Conv2D(128,kernel_size=4,strides=2,activation='relu'))
  model.add(Dropout(0.5))
```

```
  model.add(Conv2D(256,kernel_size=4,strides=1,activation='relu'))
  model.add(Conv2D(256,kernel_size=4,strides=2,activation='relu'))
  model.add(Flatten())
  model.add(Dropout(0.5))
  model.add(Dense(512,activation='relu'))
  model.add(Dense(26,activation='softmax'))
  model.summary()
  return model
from tensorflow.keras.callbacks import EarlyStopping
early_stop = EarlyStopping(monitor='val_loss',patience=2)
with strategy.scope():
  model=create_model()
  model.compile(optimizer='adam',loss='categorical_crossentropy',metrics=[tf.keras.metr
ics.Precision(),tf.keras.metrics.Recall()])
  model_hist=model.fit(X_train,Y_train,epochs=5,batch_size=32,verbose=2)
  score=model.evaluate(x=X_test,y=Y_test,verbose=0)
print('Testaccuracy:',score[1])
```

For checking the dataset: Click Here

References

- Sahoo, A. K. (2021, June). Indian sign language recognition using machine learning techniques. In *Macromolecular Symposia* (Vol. 397, No. 1, p. 2000241).
- Halder, A., & Tayade, A. (2021). Real-time vernacular sign language recognition using mediapipe and machine learning. *Journal homepage: www. ijrpr. com ISSN*, *2582*, 7421.
- Camgoz, N. C., Koller, O., Hadfield, S., & Bowden, R. (2020). Sign language transformers: Joint end-to-end sign language recognition and translation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 10023-10033).
- Moryossef, A., Tsochantaridis, I., Aharoni, R., Ebling, S., & Narayanan, S. (2020, August). Real-time sign language detection using human pose estimation. In *European Conference on Computer Vision* (pp. 237-248). Springer, Cham.
- Rajam, P. S., & Balakrishnan, G. (2011, September). Real time Indian sign language recognition system to aid deaf-dumb people. In *2011 IEEE 13th international conference on communication technology* (pp. 737-742). IEEE.
- Raghuveera, T., Deepthi, R., Mangalashri, R., & Akshaya, R. (2020). A depth-based Indian sign language recognition using microsoft kinect. *Sādhanā*, *45*(1), 1-13.
- Raheja, J. L., Mishra, A., & Chaudhary, A. (2016). Indian sign language recognition using SVM. *Pattern Recognition and Image Analysis*, *26*(2), 434-441.
- P.Likhar,N.K.BhagatandR.GN,"Deep Learning Methods for Indian Sign LanguageRecognition,"2020IEEE10thInternationalConferenceonConsumer Electronics(ICCE-Berlin),Berlin,Germany,2020,pp.1-6,doi:10.1109/ICCEBerlin50680.2020.9352194.
- P.C. Badheand V. Kulkarni, "Artificial Neural Network based Indian Sign LanguageRecognitionusinghandcraftedfeatures,"202011thInternational Conference

on Computing,Communication and Networking Technologies (ICCCNT), Kharagpur, India, 2020, pp. 1-6, doi: 10.1109/ICCCNT49239.2020.9225294.

- D. EnikeevandS. Mustafina," Recognition of Sign Language Using Leap MotionControllerData,"20202ndInternationalConferenceonControlSystems, Mathematical Modeling, Automation and Energy Efficiency(SUMMA),Lipetsk, Russia,2020,pp.393-397,doi:10.1109/SUMMA50634.2020.9280795.
- M.Al-Hammadietal.,"DeepLearning-BasedApproachforSignLanguage Gesture Recognition With Efficient Hand Gesture Representation, "in IEEE Access,vol.8,pp.192527-192542,2020,doi:10.1109/ACCESS.2020.3032140
- S.C.J.andL.A.,"Signet:A Deep Learning based Indian Sign Language RecognitionSystem,"2019InternationalConferenceonCommunicationand Signal Processing (ICCSP), Chennai, India, 2019, pp. 0596-0600, doi: 10.1109/ICCSP.2019.8698006.
- P.V.V.Kishore,D.A.Kumar,A.S.C.S.SastryandE.K.Kumar,"Motionlets MatchingWithAdaptiveKernelsfor3-DIndianSignLanguageRecognition,"in IEEESensorsJournal,vol.18,no.8,pp.3327-3337,15April15,2018,doi: 10.1109/JSEN.2018.2810449.
- M.F.Nurnoby,E.-S.M.El-AlfyandH.Luqman,"Evaluation of CNN Models with Transfer Learning for Recognition of Sign Language Alphabets with Complex Background,"2020 InternationalConference on Innovation and Intelligence for Informatics, Computing and Technologies (3ICT),Sakheer, Bahrain,2020,pp.1-6,doi:10.1109/3ICT51146.2020.9311989.
- D.Hatibaruah,A.K.TalukdarandK.Kumar Sarma,"A Static Hand Gesture Based Sign Language Recognition System using Convolutional Neural Networks, "2020IEEE17thIndiaCouncilInternationalConference(INDICON), NewDelhi,India,2020,pp.1-6,doi:10.1109/INDICON49873.2020.9342405
- M.Safeel,T.Sukumar,S.K.S,A.M.D,S.RandP.S.B," Sign Language Recognition Techniques-A Review,"2020IEEEInternationalConferencefor Innovation in Technology (INOCON),Bangluru,India,2020,pp.1-9,doi: 10.1109/INOCON50539.2020.9298376.
- A.A.Hosain,P.S.Santhalingam,P.Pathak,H.RangwalaandJ.Koecká, "FineHand:LearningHandShapesforAmericanSignLanguageRecognition," 2020 15th IEEE InternationalConference on Automatic Face and Gesture Recognition (FG 2020),Buenos Aires,Argentina,2020,pp.700-707,doi: 10.1109/FG47880.2020.00062.
- M.S.Amin,M.T.Amin,M.Y.Latif,A.A.Jathol,N.AhmedandM.I.N.Tarar, "AlphabeticalGestureRecognitionofAmericanSignLanguageusingE-Voice SmartGlove,"2020 IEEE 23rd InternationalMultitopicConference(INMIC), Bahawalpur,Pakistan,2020,pp.1-6,doi:10.1109/INMIC50486.2020.9318185.
- Y.LuandY.Lang,"Sign Language Recognition with CW Radar and Machine Learning,"202021stInternationalRadarSymposium (IRS),Warsaw,Poland, 2020,pp.31-34,doi:10.23919/IRS48640.2020.9253925.
- K.Kudrinko,E.Flavin,X.ZhuandQ.Li,"Wearable Sensor-Based Sign LanguageRecognition:AComprehensiveReview,"inIEEEReviewsinBiomedical Engineering,vol.14,pp.82-97,2021,doi:10.1109/RBME.2020.3019769.
- Y.Zhang,J.Liao,M.Ran,X.Li,S.WangandL.Liu,"ST-Xception:A Depthwise Separable Convolution Network for Military Sign Language Recognition,"2020 IEEE

International Conference on Systems, Man, and Cybernetics (SMC), Toronto, ON,Canada, 2020, pp. 3200-3205, doi: 10.1109/SMC42975.2020.9283407.

-)K.NimishaandA.Jacob,"A Brief Review of the Recent Trends in Sign LanguageRecognition,"2020InternationalConferenceonCommunicationand Signal Processing (ICCSP), Chennai, India, 2020, pp. 186-190, doi: 10.1109/ICCSP48568.2020.9182351.
- N.SarhanandS.Frintrop,"Transfer Learning For Videos: From Action RecognitionToSignLanguageRecognition,"2020IEEEInternationalConference On Image Processing (ICIP) ,Abu Dhabi, United Arab Emirate s,2020,pp.1811-1815,doi:10.1109/ICIP40778.2020.9191289.
- K.Zhao,K.Zhang,Y.Zhai,D.WangandJ.Su,"Real-Time Sign Language RecognitionBasedonVideoStream,"202039thChineseControlConference 21 (CCC), Shenyang, China, 2020, pp. 7469-7474, doi: 10.23919/CCC50068.2020.9188508.
- S.Tornay,M.RazaviandM.Magimai.-Doss,"Towards Multilingual Sign LanguageRecognition,"ICASSP2020-2020IEEEInternationalConferenceon Acoustics,SpeechandSignalProcessing(ICASSP),Barcelona,Spain,2020,pp. 6309-6313,doi:10.1109/ICASSP40776.2020.9054631.
- D.PahujaandS.Jain," Recognition of Sign Language Symbols using Templates," 2020 8thInternational Conference on Reliability, Infocom TechnologiesandOptimization(TrendsandFutureDirections)(ICRITO),Noida, India,2020,pp.1157-1160,doi:10.1109/ICRITO48877.2020.9198001.
- R.Abiyev,J.B.IdokoandM.Arslan,"ReconstructionofConvolutionalNeural NetworkforSignLanguageRecognition,"2020InternationalConferenceon Electrical,Communication,andComputerEngineering(ICECCE),Istanbul,Turkey, 2020,pp.1-5,doi:10.1109/ICECCE49384.2020.9179356
- Sridhar, A., Ganesan, R. G., Kumar, P., & Khapra, M. (2020, October). Include: A large scale dataset for indian sign language recognition. In *Proceedings of the 28th ACM international conference on multimedia* (pp. 1366-1375).
- Subramanian, B., Olimov, B., Naik, S. M., Kim, S., Park, K. H., & Kim, J. (2022). An integrated mediapipe-optimized GRU model for Indian sign language recognition. *Scientific Reports*, *12*(1), 1-16.
- Rao, G. A., & Kishore, P. V. V. (2018). Selfie video based continuous Indian sign language recognition system. *Ain Shams Engineering Journal*, *9*(4), 1929-1939.