INSTITUTE FOR ADVANCED COMPUTING

AND

SOFTWARE DEVELOPMENT

AKURDI, PUNE

Documentation On

"Sign Language Recognition"

e-DBDA MAY 2021

Submitted By:

Group No: 11

Apurva Shikhare - 1307

Pooja Dhote - 1340

Mr. Prashant Karhale
Centre Coordinator

Mr. Akshay Tilekar
Project Guide

INDEX

List of figures and Screenshots

Abstract

This project aims at identifying alphabets in Indian Sign Language from the corresponding gestures. Indian Sign Language uses both hands to make gestures instead of one hand, unlike American Sign Language. It leads to the occlusion of features and this is a major barrier to the lack of development in this field. This can be very helpful for the deaf and dumb people in communicating with others as knowing sign language is not something that is common to all, moreover, this can be extended to creating automatic editors, where the person can easily write by just their hand gestures. Also, the connecting wires restrict the freedom of movement. This system was also implemented by using Image Processing. In this way of implementation, the sign language recognition part will be done by Image Processing instead of using Gloves. In this sign language recognition project, we creating a sign detector, which detects alphabets A-Z and numbers from 0 to 9. A traditional algorithm takes some input and some logic in the form of code and drums up the output. As opposed to this, a Machine Learning Algorithm takes an input and an output and gives some logic which can then be used to work with new input to give one an output. In this classification, machine learning algorithms are trained using a set of image data. This is divided into three parts dataset, Training a CNN on the captured dataset, Predicting the data.

CHAPTER 1

INTRODUCTION

1.1 Image Processing:

Image Processing is a very powerful technique used today to convert an image into a form which is either digital or analog so that it can be used to extract some important and useful data. This process takes a raw image as an input and processes it and gives an improved modified image or characteristics associated with that image as an output. Image processing when used in ML algorithms such as CNN can be used to get very interesting results such as image recognition or creating a model to predict some feature from image. Mainly these three processes are involved in image processing. Fetching the required image by using any available tool. The fetched image is then analyzed and some necessary manipulation is done on it to find some significant patterns in it which are not visible to a naked human eye. The last stage is the output stage where the output is either an image or a report based.

1.2 Convolutional Neural Networks (CNN):

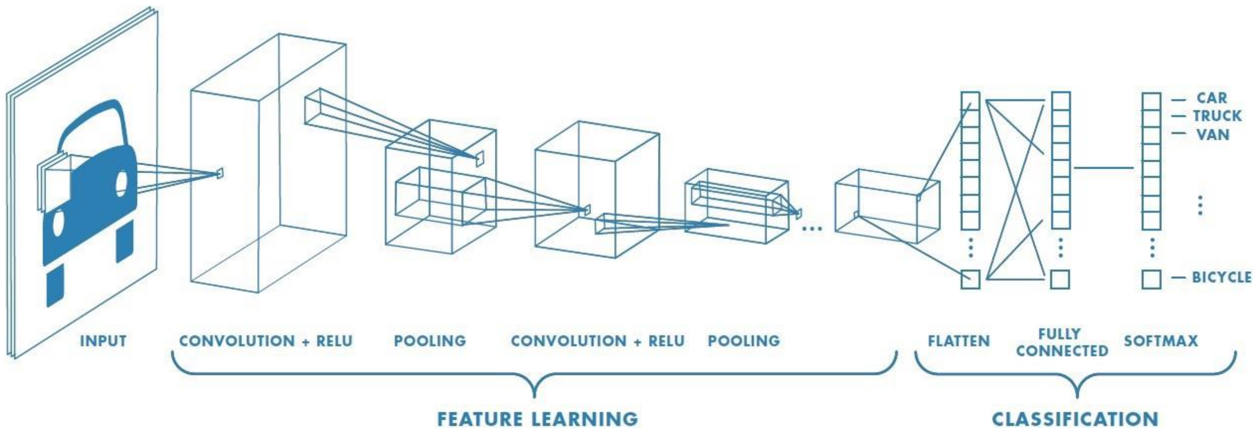


Figure 1: Convolutional Neural Network Architecture

Convolutional neural networks are distinguished from other neural networks by their superior performance with image, speech, or audio signal inputs. They have three main types of layers, which are:

- Convolutional layer
- Pooling layer
- Fully-connected (FC) layer

The convolutional layer is the first layer of a convolutional network. While convolutional layers can be followed by additional convolutional layers or pooling layers, the fully-connected layer is the final layer. With each layer, the CNN increases in its complexity, identifying greater portions of the image. Earlier layers focus on simple features, such as colours and edges. As the image data progresses through the layers of the CNN, it starts to recognize larger elements or shapes of the object until it finally identifies the intended object.

1.3 Problem Statement:

Build a model using deep learning algorithms and image processing to predict sign language by taking their image as input. Use Transfer Learning and Data Augmentation to build a model that recognizes A-Z alphabets and 0-9 numbers.

1.4 Motivation:

Communication is one of the basic requirement for survival in society. Deaf and dumb people communicate among themselves using sign language but normal people find it difficult to understand their language. Extensive work has been done on American sign language recognition but Indian sign language differs significantly from American sign language. ISL uses two hands for communicating whereas ASL uses single hand for communicating. Using both hands often leads to security of features duet overlapping of hands. In addition to this, lack of datasets along with variance in sign language with locality has resulted in restrained efforts in ISL gesture detection. Our project aims at taking the basic step in bridging the communication gap between normal people and deaf and dumb people using Indian sign language. Effective extension of this project to words and common expressions may not only make the deaf and dumb people communicate faster and easier with outer world, but also provide a boost in Developing autonomous systems for understanding and aiding them.

1.5 Scope of the project:

1.5.1 Initial functional requirement will be:

• To build deep learning algorithms to predict the sign language.

• To train the model on 35 classes of alphabets A-Z and numbers from 0-9 on over 42000 images.

• To use Transfer Learning to do feature extraction of training data and build a CNN model over it to predict sign language.

• Use Python, Keras and image processing tools to filter our data and cleaning of our input data, and use open CV to capture image and predict the class.

1.5.2 Initial nonfunctional requirement will be:

• There is data set of images of sign language. Each directory has a filename that is its name of sign language. Each directory contains images of their respective class.

CHEPTER 2

LITERATURE REVIEW

2.1 What is Machine Learning?

Machine learning is a field of Artificial Intelligence, it is an approach that is based on the idea that machines can be given access to data along with the ability to learn from it. Machine can find patterns in the data and predict according to that. The data provided is divided to training and testing data and the accuracy of the algorithm is calculated on the basis of test score it obtains. Machine Learning is mainly divided into 3 categories:

a. **Supervised Machine Learning:**

   Supervised Learning is a type of machine learning used to train models from labeled training data. It allows you to predict output for future or unseen data.
   In supervised learning, algorithm is selected based on target variable
   Types Of Supervised learning are

   1. Classification:
      If target variable is categorical (classes), then use classification algorithm.
      classification is applied when the output has finite and discrete values.
   2. Regression:
      If target variable is a continuous numeric variable (100–2000), then use a regression algorithm.

b. **Unsupervised Machine Learning:**

   This kind of algorithm is used when the input data is not classified as well as not labeled. These algorithms identify the patterns in the provided data and draw inferences from the data sets to identify the data which is unstructured and unlabeled. However, these algorithms are not capable of predicting very accurate results but still it provides quite efficient results base on the kind of unstructured data provided. These algorithms are more complex and difficult to understand than the supervised algorithms.

c. **Reinforcement Machine learning:**

   Reinforcement learning is an area of machine learning it is generally depending on users' feedback how user can react on specific area or product. It recommends user upon their previous activity. in reinforcement learning, there is no answer but the reinforcement agent decides what to do to perform the given task. In the absence of a training dataset, it is bound to learn from its experience.

2.2 Transfer Learning:

Transfer learning generally refers to a process where a model trained on one problem is used in some way on a second related problem. In deep learning, transfer learning is a technique whereby a neural network model is first trained on a problem similar to the problem that is being solved. One or more layers from the trained model are then used in a new model trained on the problem of interest. Transfer learning has the benefit of decreasing the training time for a neural network model and can result in lower generalization error. The weights in re-used layers may be used as the starting point for the training process and adapted in response to the new problem. This usage treats transfer learning as a type of weight initialization scheme. This may be useful when the first related problem has a lot more labeled data than the problem of interest and the similarity in the structure of the problem may be useful in both contexts.

CHAPTER 3

SYSTEM DEVELOPMENT

3.1 System Environment:

System environment involves a process of designing a software prototype, testing it, converting it to complete model and again applying various testing algorithms to it and finally create the whole software.

3.2 Functional requirement specification:

Use Cases:

- To Use Transfer Learning for feature Extraction.
- To build deep learning algorithms and predict the sign language.

Brief Description:

The neural network then identifies the patterns in that matrix to remember that image so that it can later itself recognize that image. This neural network will take the value of the pixels as a feature on which the ML algorithm will work to predict the output. It is almost impossible for a human eye to recognize the pattern which this network learns to identify that image. We can provide different weights and bias to the model to alter our results.

Initial Step by Step Description:

- First, we need to import the images then perform filtering and data augmentation.
- We are going to do feature extraction using pretrained models.
- Then we are going to train a deep learning model based on these extracted features for better accuracy.

3.3 Non-functional requirement:

Python libraries such as OS, TensorFlow, numpy, OpenCV, Matplotlib, Keras.

CHAPTER 4

REQUIREMENT SPECIFICATIONS

4.1 External requirement specification:

The only link to an external system is the link to the image with customer. The Administrator believes that a site member is much more likely to be an effective reviewer and has imposed a membership requirement for a Reviewer.

4.2 Detailed Non-Functional Requirements:

4.2.1 Functional Requirement:

First of all model should be successfully trained by developer.

Download and install Anaconda (windows version).

4.2.2 Hardware Requirement:

- **Processor:** Intel Dual Core
- **RAM:** Minimum 1GB
- **OS:** Windows,Linux.MacOS

4.2.3 Software Requirement:

Anaconda Navigator:

Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda® distribution that allows you to launch applications and easily manage conda packages, environments and channels without using command-line commands. It is available for Windows, macOS and Linux.

Following libraries of python should be install:

1. **OpenCV –** pip install opencv-python (hand detection).
2. **TensorFlow –** pip install tensorflow (keras uses TensorFlow as backend).
3. **Keras –** pip install keras (to build our classification model).

OpenCV-Python:

OpenCV python is an appropriate tool for fast prototyping of computer vision problems. Compared to other languages like C/C++, Python is slower. But another important feature of Python is that it can be easily extended with C/C++. This feature helps us to write computationally intensive codes in C/C++ and create a Python wrapper for it so that we can use these wrappers as Python modules. This gives us two advantages: first, our code is as fast as original C/C++ code (since it is the actual C++ code working in background) and second, it is very easy to code in Python. This is how OpenCV-Python works, it is a Python wrapper around original C++ implementation. And the support of Numpy makes the task more easier. Numpy is a highly optimized library for numerical operations. It gives a MATLAB-style syntax. All the OpenCV array structures are converted to-and-from Numpy arrays. So whatever operations you can do in Numpy, you can combine it with OpenCV, which supports Numpy can be used with this. So OpenCV-Python is an appropriate tool for fast prototyping of computer vision problems.

Tensorflow:

A couple of years ago, deep learning started to outperform all other machine learning algorithms when giving a massive amount of data. Google saw it could use these deep neural networks to improve its services:

- Gmail
- Photo
- Google search engine

They build a framework called Tensorflow to let researchers and developers work together on an AI model. Once developed and scaled, it allows lots of people to use it. It was first made public in late 2015, while the first stable version appeared in 2017. It is open source under Apache Open Source license. You can use it, modify it and redistribute the modified version for a fee without paying anything to Google.

TensorFlow Architecture (Tensorflow architecture works in three parts**):**

- Preprocessing the data
- Build the model
- Train and estimate the model

CHAPTER 5

SYSTEM DESIGN

5.1 Flowchart of the System:

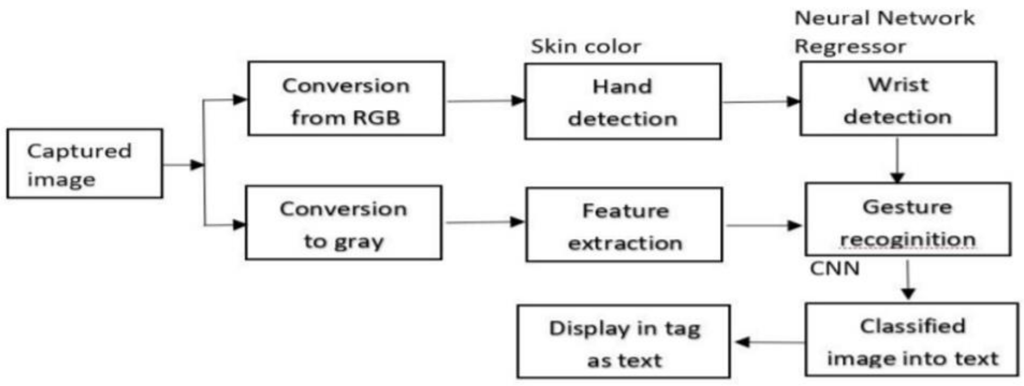


Figure 2: Image Processing Flowchart

Steps involved in flow-chart:

1. The dataset is imported from the web and is broken into three parts i.e. training, validation and testing.
2. Reading all Sign images from training dataset.
3. Now based on the training dataset and validation dataset the model is trained and validated. The weights are adjusted based on back propagation .
4. Take picture of the hand to be tested using a webcam.
5. Convert the image into Grayscale.
6. Convert the Grayscale image into a binary image. Set a threshold so that the pixels that are above certain intensity are set to white and those below are set to black.
7. From the binary image, we generate the coordinates of the image.
8. These coordinates are then compared with stored co-ordinates in the database for the purpose of output generation using pattern matching technique.
9. We need to use a pattern matching algorithm for this purpose.
10. If the pattern is matched, the alphabet corresponding to the image is displayed
11. We have 35 classes so we have to use label encoding and set the position of the data.

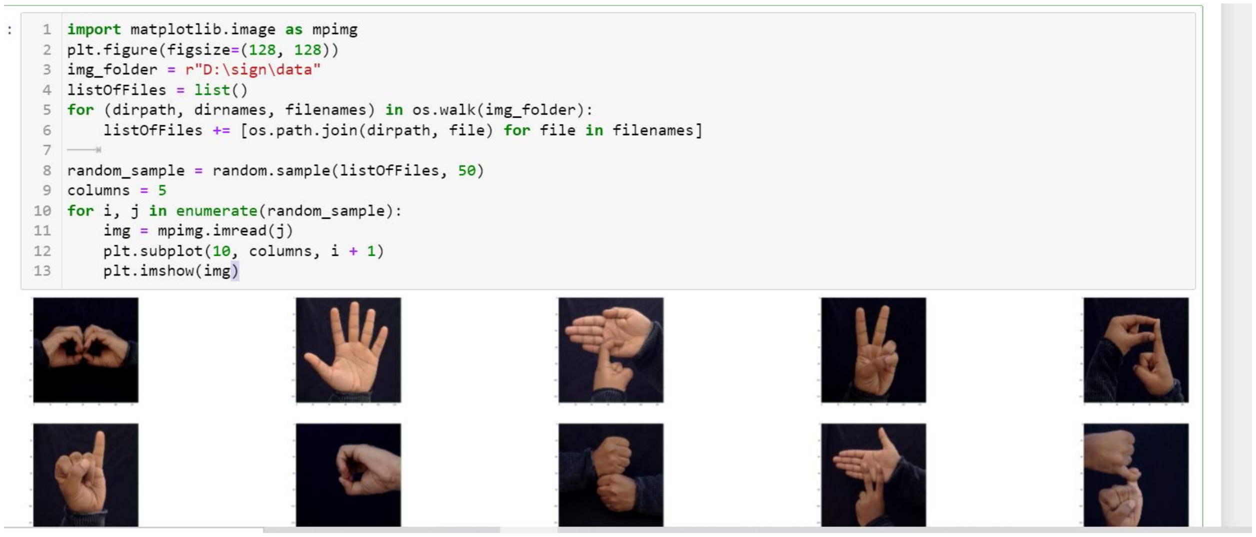
CHAPTER 6

MODEL BUILDING

6.1 Data Preprocessing:

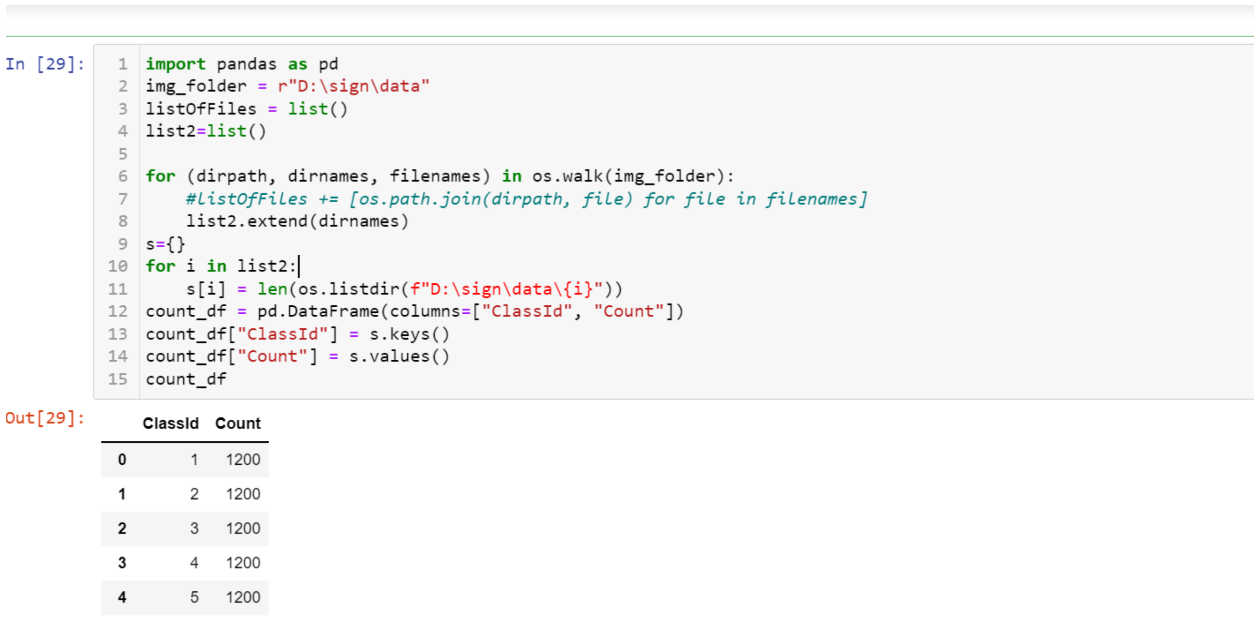
The datasets consist of folders of color images, to split dataset into train and test, we import splitfolders and each datasets contains no of folders naming as their classes name. Folder contains no of images of hand gestures so for reading that image for specific folder. It is somehow critical to read all images for each folder so we use library for reading those data. For understanding data, we use os module to check no of classes in the data and then check total number of images in each dataset, os is used to interact with operating system. There are 35 types of classes present in each train and test.

Total No of images in train datasets are 33566 and 8400 no of images in test datasets.



```
import matplotlib.image as mpimg
plt.figure(figsize=(128, 128))
img_folder = r"D:\sign\data"
listOfFiles = list()
for (dirpath, dirnames, filenames) in os.walk(img_folder):
    listOfFiles += [os.path.join(dirpath, file) for file in filenames]

random_sample = random.sample(listOfFiles, 50)
columns = 5
for i, j in enumerate(random_sample):
    img = mpimg.imread(j)
    plt.subplot(10, columns, i + 1)
    plt.imshow(img)
```



In [29]:

```python
import pandas as pd
img_folder = r"D:\sign\data"
listOfFiles = list()
list2=list()

for (dirpath, dirnames, filenames) in os.walk(img_folder):
    #listOfFiles += [os.path.join(dirpath, file) for file in filenames]
    list2.extend(dirnames)
s={}
for i in list2:
    s[i] = len(os.listdir(f"D:\sign\data\{i}"))
count_df = pd.DataFrame(columns=["ClassId", "Count"])
count_df["ClassId"] = s.keys()
count_df["Count"] = s.values()
count_df
```

Out[29]:

	ClassId	Count
0	1	1200
1	2	1200
2	3	1200
3	4	1200
4	5	1200

Figure 3: Data preprocessing

6.2 Data Augmentation:

Data augmentation in data analysis are techniques used to increase the amount of data by adding slightly modified copies of already existing data or newly created synthetic data from existing data. It acts as a regularizer and helps reduce overfitting when training a machine learning model. The ImageDataGenerator class supports a number of pixel scaling methods, as well as a range of data augmentation techniques. We will focus on the pixel scaling techniques. The ImageDataGenerator class can be used to rescale pixel values from the range of 0-255 to the range 0-1 preferred for neural network models. Here we normalize the size of image. So it can be easy for further processing of the data. Rescale is a value by which we will multiply the data before any other processing. Our original images consist in RGB coefficients in the 0-255, but such values would be too high for our model to process (given a typical learning rate), so we target values between 0 and 1. We used a batch size of 35. This means that each of the train and test datasets of images are divided into groups of 35 images that will then be scaled when returned from the iterator. Batch size is simply No. of images to be yielded from the generator per batch.

6.3 Data Modeling:

To build a model in keras we used sequential model, it is the easiest way to build a model in Keras. It allows you to build a model layer by layer. Each layer has weights that correspond to the layer the follows it. We use the 'add()' function to add layers to our model. We will add two layers and an output layer.

Convolutional Layer:

The convolutional layer is the core building block of a CNN, and it is where the majority of computation occurs. It requires a few components, which are input data, a filter, and a feature map. Let’s assume that the input will be a color image, which is made up of a matrix of pixels in 3D. This means that the input will have three dimensions—a height, width, and depth—which correspond to RGB in an image. We also have a feature detector, also known as a kernel or a filter, which will move across the receptive fields of the image, checking if the feature is present. This process is known as a convolution. Convolutional layer converts the image into numerical values, allowing the neural network to interpret and extract relevant patterns.

Activation Function:

Activation functions are mathematical equations that determine the output of a neural network. The function is attached to each neuron in the network, and determines whether it should be activated or not, based on whether each neuron’s input is relevant for the model prediction. In simple words, it is a transformation function that maps the input signals into output signals that are needed for the neural network to function. Activation functions help us to determine the output of the program by normalizing the output values in a range of 0 to 1.

In our case, we have mostly used a SoftMax activation function. The SoftMax regression is a form of logistic regression that normalizes an input value into a vector of values that follows a probability distribution whose total sums up to 1. The output values are between the range [0,1] which is nice because we are able to avoid binary classification and accommodate as many classes or dimensions in our neural network model.

Pooling Layer:

Pooling layers, also known as downsampling, conducts dimensionality reduction, reducing the number of parameters in the input. Similar to the convolutional layer, the pooling operation sweeps a filter across the entire input, but the difference is that this filter does not have any weights. Instead, the kernel applies an aggregation function to the values within the receptive field, populating the output array. There are two main types of pooling:

- **Max pooling:** As the filter moves across the input, it selects the pixel with the maximum value to send to the output array. As an aside, this approach tends to be used more often compared to average pooling.
- **Average pooling:** As the filter moves across the input, it calculates the average value within the receptive field to send to the output array.

While a lot of information is lost in the pooling layer, it also has a number of benefits to the CNN. They help to reduce complexity, improve efficiency, and limit risk of overfitting.

Stride and Padding:

Stride is the number of pixels shifts over the input matrix. When the stride is 1 then we move the filters to 1 pixel at a time. Stride and Padding are the two terms which play an important role in the convolution neural network. Stride is a parameter of the neural network's filter that modifies the amount of movement over the image. Padding is the space between an image or cell contents and its outside border. Padding specifies whether we have to add extra bits on the corner of an image or not.

Dropout Layer:

Dropout refers to dropping or removing of neurons at random from a hidden layer in neural network to avoid overfitting. Dropout is an approach to regularization in neural network which helps in independent learning of the neurons. It is highly effective in avoid overfitting. Regularization reduces overfitting by adding penalty to all features, which are useless.
Dropout Forces the Neural Network to learn more Robust Features that are Useful for Predictive analytics. Dropout Helps us to Reduce the Training Time required for the Neural Network. Choosing the Right value for Dropout is crucial to get Good Results.

Flatten Layer:

Flatten is the function that converts the pooled feature map to a single column that is passed to the fully connected layer. Dense adds the fully connected layer to the neural network.

Dense Layer:

Dense Layer is a type of layer in which will all the input of current layer is connected to the next layer. In our model we've used dense layer at the end of model which will output 35 by calculating the probability of a certain species, the one with maximum probability is our bird corresponding to that image.

Feed Forward:

Feed Forward in a model means that we pass an input to the neural network in our case it is convolution neural network, the output is predicted based on the input and intermediate weights are initialized using random distributions. Once we predicted the output, we will calculate the error function by comparing the predicted output with the actual output. Based on this error we back propagate to reduce the error. This error can of several types like mean squared error, root mean squared error, categorical cross-entropy etc. There's a lot of calculation involved in this feed forward and back propagation.

Backpropagation:

Backpropagation is a technique of optimization of error function by using various optimization techniques like stochastic gradient descent, adagrad, rmsprop, Adam etc. The main purpose is to reduce by adjusting the weights of intermediate layers which requires a lot of calculations. In our case, we have used rmsprop optimizer to reduce the error function and increase the accuracy of our model.

Fully-Connected Layer:

The name of the full-connected layer aptly describes itself. As mentioned earlier, the pixel values of the input image are not directly connected to the output layer in partially connected layers. However, in the fully-connected layer, each node in the output layer connects directly to a node in the previous layer.While convolutional and pooling layers tend to use ReLu functions, FC layers usually leverage a softmax activation function to classify inputs appropriately, producing a probability from 0 to 1**.**

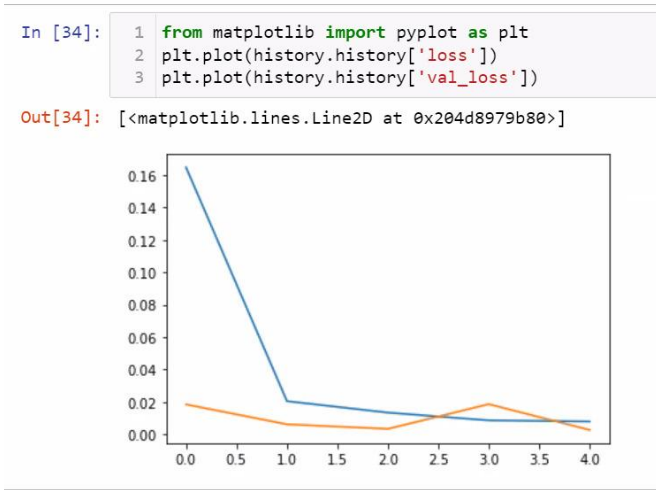
Image Segmentation:

Image segmentation is a branch of digital image processing which focuses on partitioning an image into different parts according to their features and properties. The primary goal of image segmentation is to simplify the image for easier analysis. In image segmentation, you divide an image into various parts that have similar attributes. The parts in which you divide the image are called Image Objects.

Predict the Gesture:

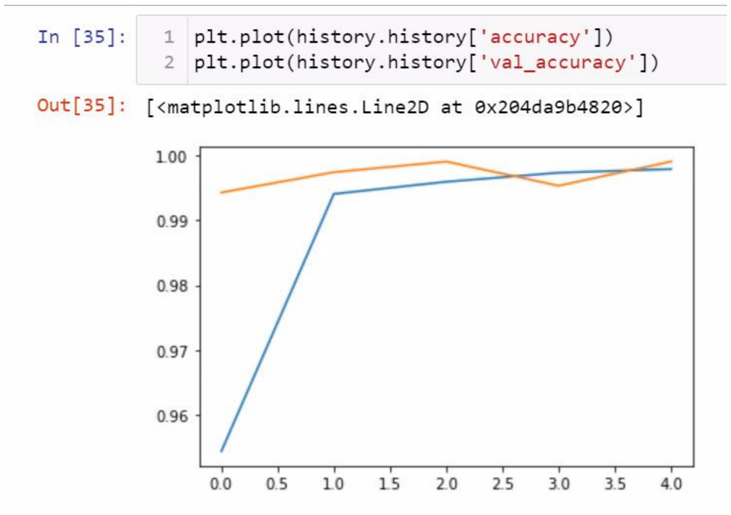
In this, we create a bounding box for detecting the ROI and calculate the accumulated_avg. This is done for identifying any foreground object**.** Now we find the max contour and if contour is detected that means a hand is detected so the threshold of the ROI is treated as a test image. Feed the threshold image of the ROI consisting of the hand as an input to the model for prediction. Now we load the model that we had created earlier and set some of the variables that we need, i.e, initializing the background variable, and setting the dimensions of the ROI. Segmenting the hand, i.e, getting the max contours and the thresholded image of the hand detected.

Accuracy and Loss Plot:



```
In [34]:  from matplotlib import pyplot as plt
          plt.plot(history.history['loss'])
          plt.plot(history.history['val_loss'])

Out[34]: [<matplotlib.lines.Line2D at 0x204d8979b80>]
```



```
In [35]:  plt.plot(history.history['accuracy'])
          plt.plot(history.history['val_accuracy'])

Out[35]: [<matplotlib.lines.Line2D at 0x204da9b4820>]
```

Figure 4: Plot of Accuracy and Loss

Description:

Here we plot graph of accuracy of train and validation data. By observing graph we can say that accuracy of model is very good. Here we get 99.83% accuracy on the test data. so we say that our model of CNN is good fit for the data. This can be further extended for detecting the Hindi words.

CHAPTER 7

RESULTS

The output of the sign language will be displayed in the text form in real time. This makes the system more efficient and hence communication of the hearing and speech impaired people more easy. The images captured through web cam are compared and the result of comparison is displayed at the same time. Thus this feature of the system makes communication very simple and delay free.

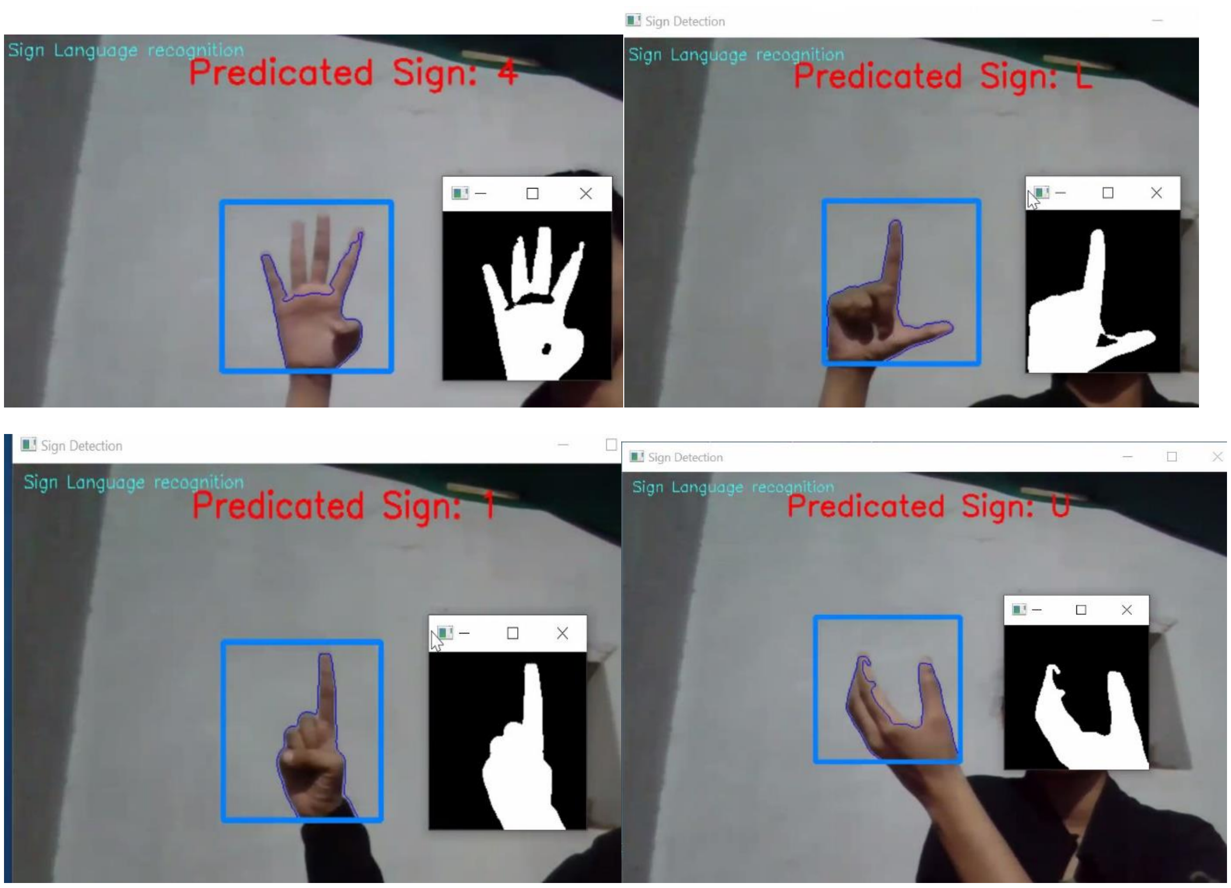


Figure 5: Prediction Results

FUTURE SCOPE

- Image Processing part should be improved so that System would be able to communicate in both directions i.e.it should be capable of converting normal language to sign language and vice versa. We will try to recognize signs which include motion. Moreover we will focus on converting the sequence of gestures into text i.e. word and sentences and then converting it into the speech which can be heard.
- We are planning to achieve higher accuracy even in case of complex backgrounds by trying out various background subtraction algorithms.
- We are also thinking of improving the preprocessing to predict gestures in low light conditions with a higher accuracy.

CONCLUSION

Many breakthroughs have been made in the field of artificial intelligence, machine learning and computer vision. They have immensely contributed in how we perceive things around us and improve the way in which we apply their techniques in our everyday lives. Many researches have been conducted on sign gesture recognition using different techniques like ANN, LSTM and 3D CNN. However, most of them require extra computing power . On the other hand, our project requires low computing power and gives a remarkable accuracy of above 90%. In our project, we proposed to normalise and rescale our images to 64 pixels in order to extract features (binary pixels) and make the system more robust. We use CNN to classify the A-Z alphabets and numbers from 0 to 9 in Indian sign gestures and successfully achieve an accuracy of 99% which is better accuracy.

REFERENCES

[1] Joyeeta Singha, Karen Das). "Indian Sign Language Recognition Using Eigen Value Weighted Euclidean Distance Based Classification Technique ,"International Journal of Advanced Computer Science and Applications, Vol. 4, No. 2, 2013 .

[2] Shweta S. Patil1,Mr .G.P.Jain2. "Sign Language converter for deaf and dumb people", International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056 Volume: 06 Issue: 03 | Mar 2019

[3] Sanil Jain K.V.Sameer Raja. "Indian Sign Language Character Recognition", Indian Institute of Technology, Kanpur.

[4] https://www.programcreek.com/python/example/89428/cv2.absdiff

[5].https://docs.opencv.org/3.4.15/d3/dc0/group__imgproc__shape.html#gga4303f45752694956 374734a03c54d5ffa5f2883048e654999209f88ba04c302f5

[6] Aditya Das1 , Shantanu Gawde1 , Khyati Suratwala1 and Dr. Dhananjay Kalbande, "Sign Language Recognition Using Deep Learning on Custom Processed Static Gesture Images", Department of Computer Engineering Sardar Patel Institute of Technology Mumbai, India