FINAL YEAR PROJECT REPORT

at

Sathyabama Institute of Science and Technology (Deemed to be University)

Submitted in partial fulfilment of the requirements for the award of Bachelor of Engineering Degree in Computer Science and Engineering

Pitambara Awadhesh

(Reg. No. 38110406)

Reema Rose Toppo

(Reg. No. 38110458)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING SCHOOL OF COMPUTING

SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY JEPPIAAR NAGAR, RAJIV GANDHI SALAI,

CHENNAI – 600119, TAMILNADU

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A" by NAAC
(Established under Section 3 of UGC Act, 1956)
JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI– 600119
www.sathyabamauniversity.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **Pitambara Awadhesh (38110406) and Reema Rose Toppo (38110458)** who carried out the project entitled **Indian Sign Language Recognition System to Help Mute and Deaf People** under my supervision from _____ to _____.

Internal Guide
(MS. DEEPA, DR. BEVISH)

External Guide
(DR. BEVISH)

Head of the Department
(DR.L. LAKSHMANAN and DR.S. VIGNESHWARI)

Submitted for Viva-voce Examination held on _____

Internal Examiner

External Examiner

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to Board of Management of SATHYABAMA for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to Dr. T. Sasikala M.E., Ph.D., Dean, School of Computer Science and Technology and Dr. L. Lakshmanan, Head of the Department, Dept. of Computer Science and Technology for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide Dr. B. Bharathi for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the Department of Computer Science and Engineering who were helpful in many ways for the completion of the project.

INDIAN SIGN LANGUAGE RECOGNITION SYSTEM TO HELP DEAF AND MUTE PEOPLE

Submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering / Technology degree in Computer Science and Engineering (Specialisation of degree)

By

Pitambara Awadhesh (38110406)

Reema Rose Toppo (38110458)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A" by NAAC
JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600119

MAY - 2022

ABSTRACT

Everything has changed due to the COVID19 pandemic. We went from offline to internet mode in no time. Some people found it easy to adjust to this way of life, but many others with disabilities never did and still find it difficult to explain their ideas in online meetings. To solve this problem, we have proposed a Convolutional Neural Network (CNN) based model for Indian sign language recognition. In our proposed method, a mute or deaf person can interact with the camera integrated into a computer and use gestures that will be recognized and converted to text for others to understand. For this, we have created a sample dataset and pre-processed it using a label binarizer. Afterward, feature extraction was done using two models, first for the palm region and the other for the fingers. Later on, this dataset was fed into a custom-made CNN model whose learning parameters were provided as 0.001 learning rate, 128 batch-size, and 10 epochs. The model performed well with 93% of accuracy while recognizing a hand gesture. Hence, our proposed model can be integrated into other online meeting sites for the target audience to use.

CONTENTS

CHAPTER 1 - INTRODUCTION

1.1 Basics

The goal of this project was to build a neural network able to classify which word of the Indian Sign Language (ISL) is being signed, given an image of a signing hand. This project is a first step towards building a possible sign language translator, which can take communications in sign language and translate them into written and oral language. Such a translator would greatly lower the barrier for many deaf and mute individuals to be able to better communicate with others in day-to-day interactions.

This goal is further motivated by the isolation that is felt within the deaf community. Loneliness and depression exist at higher rates among the deaf population, especially when they are immersed in a hearing world. Large barriers that profoundly affect life quality stem from the communication disconnect between the deaf and the hearing. Some examples are information deprivation, limitation of social connections, and difficulty integrating in society.

Most research implementations for this task have used depth maps generated by the depth camera and high-resolution images. The objective of this project was to see if neural networks are able to classify signed ISL letters using an input video sequence of a person taken with a personal device such as a laptop webcam. This is in alignment with the motivation as this would make a future implementation of a real-time ISL-to-oral/written language translator practical in everyday situations.

1.2 Python3

Python is an interpreted, high, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and

large-scale projects. Python is dynamically typed and garbage-collected. It supports multiple programming paradigms including structured (particularly, procedural), object-oriented, and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library.

1.3 Analysis and Visualization of Data

(a) NumPy

NumPy is a python library used for working with arrays. It also has functions for working in the domain of linear algebra, Fourier transform, and matrices. NumPy was created in 2005 by Travis Oliphant. It is an open-source project and you can use it freely. NumPy stands for Numerical Python.

Why NumPy?

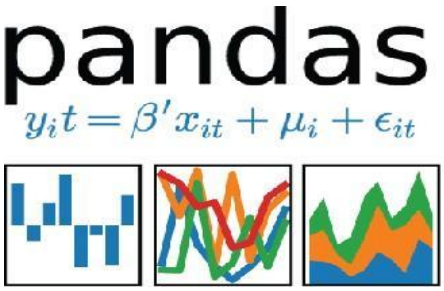
In Python, we have lists that serve the purpose of arrays, but they are slow to process. NumPy aims to provide an array object that is up to 50x faster than traditional Python lists.

Figure 1: NumPy

The array object in NumPy is called ndarray, in
provides a lot of supporting functions that make working with ndarray very easy. Arrays are very frequently used in data science, where speed and resources are very important.

(b) Pandas

Pandas is a high-level data manipulation tool developed by Wes McKinney. It is built on the Numpy package and its key data structure is called the DataFrame. DataFrames allow you to store and manipulate tabular data in rows of observations and columns of variables.



	BandName	WavelengthMax	WavelengthMin
0	CoastalAerosol	450	430
1	Blue	510	450
2	Green	590	530
3	Red	670	640
4	NearInfrared	880	850
5	ShortWaveInfrared_1	1650	1570
6	ShortWaveInfrared_2	2290	2110
7	Cirrus	1380	1360

Figure 2: Pandas: DataFrames

DataFrame object for data manipulation with integrated indexing.

- Tools for reading and writing data between in-memory data structures and different file formats.
- Data alignment and integrated handling of missing data.
- Reshaping and pivoting of data sets.
- Label-based slicing, fancy indexing, and subsetting of large data sets.
- Data structure column insertion and deletion.
- Group by engine allowing split-apply-combine operations on data sets.
- Data set merging and joining.
- Hierarchical axis indexing to work with high-dimensional data in a lower-dimensional data structure.
- Time series-functionality: Date range generation and frequency conversion, moving window statistics, moving window linear regressions, date shifting, and lagging.
- Provides data filtration.

(c) MatPlotlib

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib produces publication-quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shell, web application servers, and various graphical user interface toolkits.

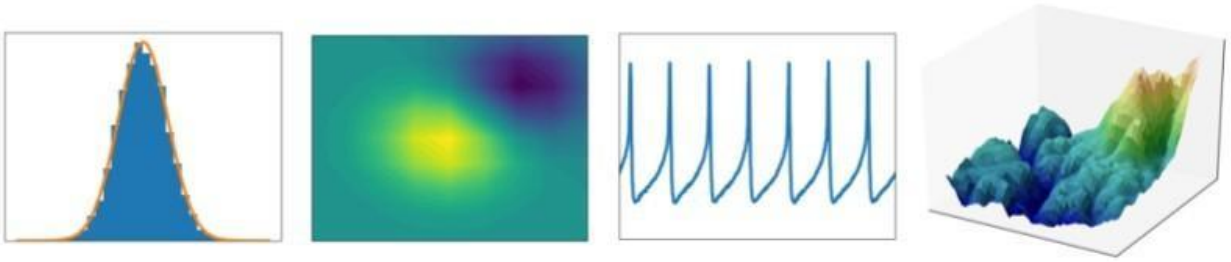


Figure 3: Plots using Matplotlib

• Convenient views onto the overall structure of complex data-set

- High-level abstractions for structuring multi-plot grids that let you easily build complex visualizations
- Concise control over matplotlib figure styling with several built-in themes
- Tools for choosing color palettes that faithfully reveal patterns in your data

1.4 Deep Learning

Deep learning is a subset of machine learning, which is essentially a neural network with three or more layers. These neural networks attempt to simulate the behavior of the human brain—albeit far from matching its ability—allowing it to “learn” from large amounts of data. While a neural network with a single layer can still make approximate predictions, additional hidden layers can help to optimize and refine for accuracy.

Deep learning drives much artificial intelligence (AI) applications and services that improve automation, performing analytical and physical tasks without human intervention.

Deep neural networks consist of multiple layers of interconnected nodes, each building upon the previous layer to refine and optimize the prediction or categorization. This progression of computations through the network is called forward propagation. The input and output layers of a deep neural network are called *visible* layers. The input layer is where the deep learning model ingests the data for processing, and the output layer is where the final prediction or classification is made.

To state the layer in a clear manner:

1. Input layer – The input layer has input features in a dataset that is known to us.
2. Hidden Layer – Hidden layer, just like we need to train the brain through hidden neurons.
3. Output layer – value that we want to classify.

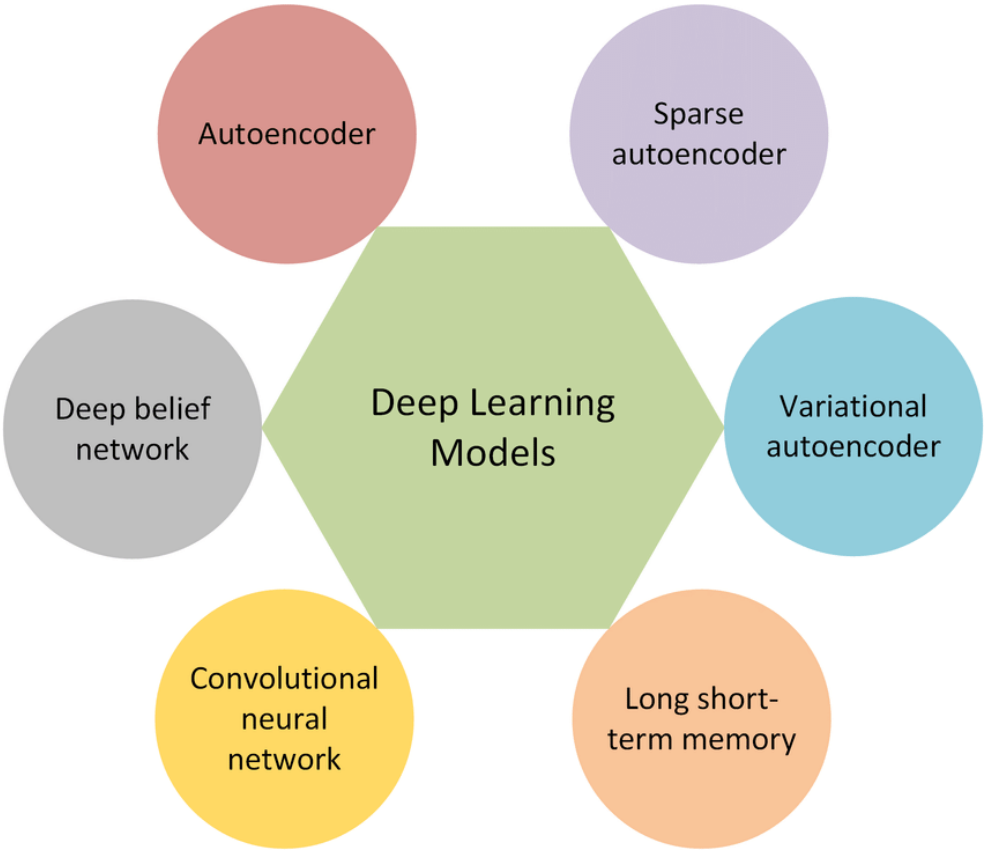


Fig 5: Types of Deep learning algorithms

There are many types of deep learning algorithms developed over the years but there are a few algorithms that are frequently used:

1. Artificial Neural Network:

An artificial Neural Network is the component of a computing system designed in such a way that the human brain analyses and makes a decision. Ann is the building block of deep learning and solves problems that seem impossible or very difficult to humans.

Artificial neural networks work like a human brain. The human brain has billions of neurons and each neuron is made up of a cell body that is responsible for computing information by carrying forward information towards hidden neurons and providing the final Output.

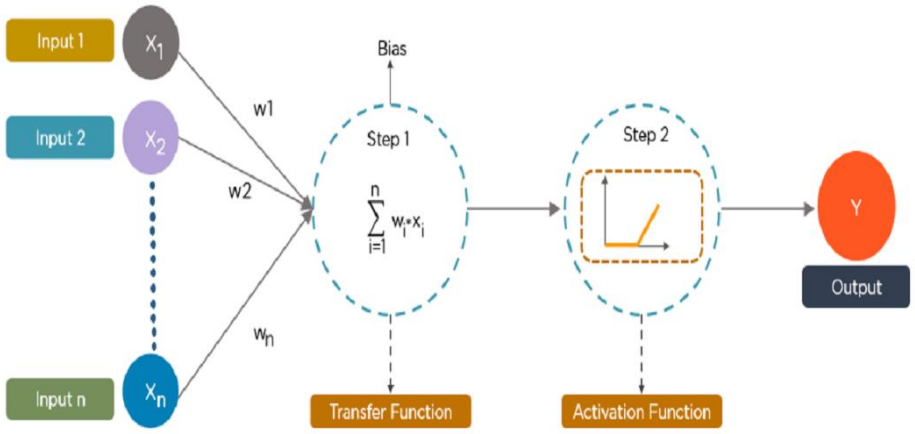


Fig 6: Artificial Neural Networks

The aim is to minimize the error by adjusting the weight and bias of the interconnection which is known as backpropagation. With the process of backpropagation, the difference between the desired output and actual output produces the least error.

2. Convolutional Neural Network

CNN is a supervised type of Deep learning, most preferable used in image recognition and computer vision. CNN has multiple layers that process and extract important features from the image. Convolutional neural networks are composed of multiple layers of artificial neurons. Artificial neurons, a rough imitation of their biological counterparts, are mathematical functions that calculate the weighted sum of multiple inputs and output an activation value. When you input an image in a ConvNet, each layer generates several activation functions that are passed on to the next layer.

The first layer usually extracts basic features such as horizontal or diagonal edges. This output is passed on to the next layer which detects more complex features such as corners or combinational edges. As we move deeper into the network it can identify even more complex features such as objects, faces, etc.

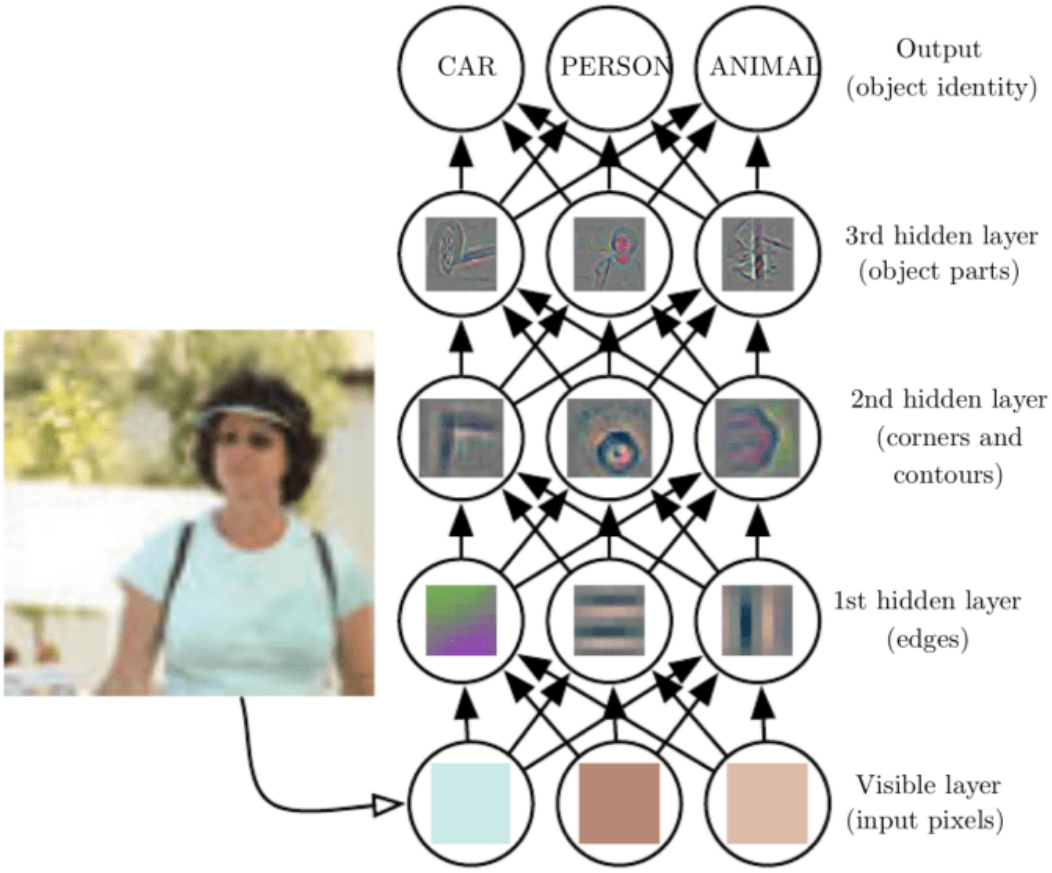


Fig 7: Working of a CNN

3. Recurrent Neural Networks (RNNs)

RNN is a type of supervised deep learning where the output from the previous step is fed as input to the current step. RNN deep learning algorithm is best suited for sequential data. RNN is most preferably used in image captioning, time-series analysis, natural-language processing, handwriting recognition, and machine translation.

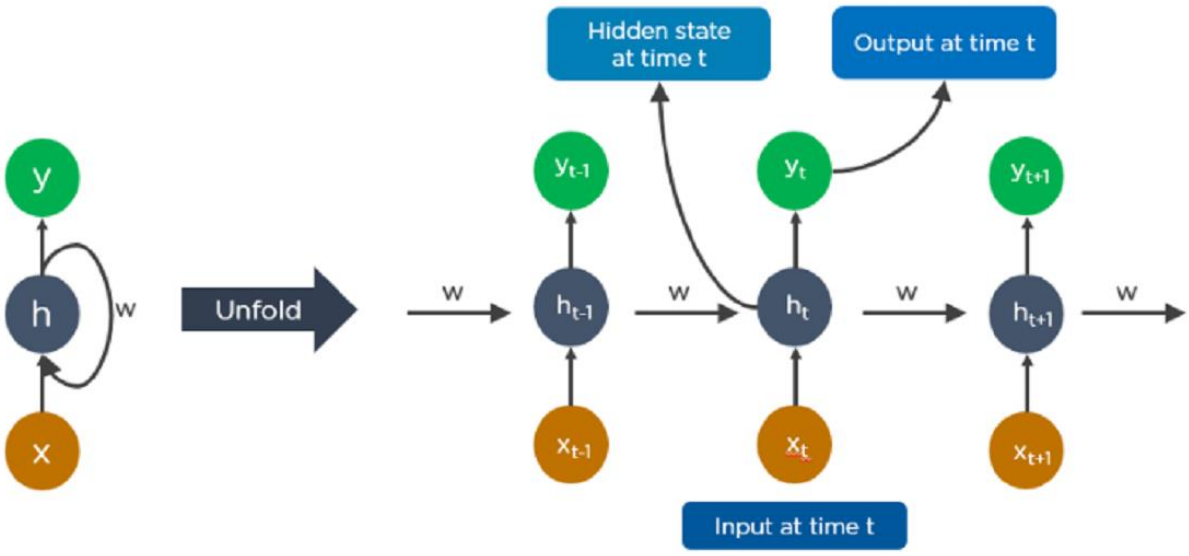


Fig 8: Recurrent Neural Networks (RNNs)

The most vital feature of RNN is the Hidden state, which memorizes some information about a sequence. There are mainly 4 steps of how RNN works.

1. The output of the hidden state at t-1 is fed into input at time t.
2. Same way, the output at time t fed into the input at time t+1.
3. RNN can process inputs of any considerable length
4. The RNN computation depends on historical sequence data and the model size doesn't increase with input size.

1.5 OpenCV

OpenCV is a great tool for image processing and performing computer vision tasks. It is an open-source library that can be used to perform tasks like face detection, objection

tracking, landmark detection, and much more. It supports multiple languages including python, java C++. Although, For this article, we will be limiting to python only.

The library is equipped with hundreds of useful functions and algorithms, which are all freely available to us. Some of these functions are really common and are used in almost every computer vision task. Whereas many of the functions are still unexplored and haven't received much attention yet.

With the help of Open CV in python, it's possible to process images, videos easily and can extract useful information from them, as there are lots of functions available. Some of the common applications are

1. Image Processing:

   Images can be read, written, shown, and processed with the OpenCV, which can generate a new image from that either by changing its shape, color, or extracting something useful from the given one and write into a new image.

2. Face Detection:

   Either from the live streaming using a web camera or from the locally stored videos/images utilizing Haar-Cascade Classifiers.

3. Face Recognition:

   It is followed by face detection from the videos using OpenCV by drawing bounding boxes i.e. rectangles and then model training using ML algorithms to recognize faces.

4. Object Detection:

Open CV along with the YOLO, an object detection algorithm can be used to detect objects from the image, videos either moving or stationary objects.

1.6 TensorFlow

Fig 9: TensorFlow

TensorFlow is an open-source library developed by Google primarily for deep learning applications. It also supports traditional machine learning. TensorFlow was originally developed for large numerical computations without keeping deep learning in mind. However, it proved to be very useful for deep learning development as well, and therefore Google open-sourced it.

TensorFlow accepts data in the form of multi-dimensional arrays of higher dimensions called tensors. Multi-dimensional arrays are very handy in handling large amounts of data.

TensorFlow works on the basis of data flow graphs that have nodes and edges. As the execution mechanism is in the form of graphs, it is much easier to execute TensorFlow code in a distributed manner across a cluster of computers while using GPUs.

1.6.1 Tensorflow Object Detection API

Creating accurate machine learning models capable of localizing and identifying multiple objects in a single image remains a core challenge in computer vision. The TensorFlow Object Detection API is an open-source framework built on top of TensorFlow that makes it easy to construct, train and deploy object detection models.

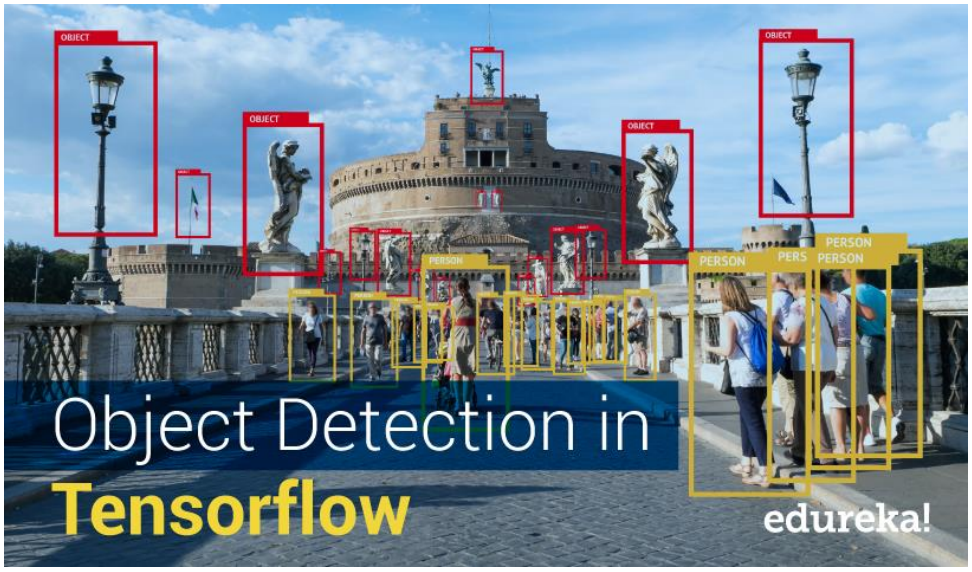


Fig 10: Object Detection in Tensorflow

CHAPTER 2 – LITERATURE SURVEY

2.1 Vision-based Real-Time Hand Gesture Recognition Techniques for Human-Computer Interaction (IEEE)

At first, we started studying the most basic human-computer interaction and how it actually works. Ghotkar and Kharate explained three techniques and experimented with interaction with a Desktop/Laptop with static hand gestures. All these techniques used a real-time approach with different feature descriptors such as Fourier Descriptor(FD),7 Hu moments, Convex Hull, and Finger Detection. Real-time Recognition efficiency was calculated with respect to recognition time for FD and 7 Hu moments.

There are two major approaches for hand gesture recognition: Data Glove, Vision-based. Each approach is having its limitations and advantages, but vision-based approaches are more feasible as compared to data gloves as users do not need to wear cumbersome devices like data gloves.

Vision-based hand gesture recognition is having challenges such as variable lighting conditions, dynamic background, and skin color detection, considering these fact algorithms were developed to overcome some of these challenges. Recognition of hand gestures executes windows applications such as opening Notepad, Windows media player, Internet Explorer, and MS-Paint. All algorithms were working on bare hands where the user need not wear any color gloves or data gloves.

2.2 Real-Time Hand Gesture Recognition Using Finger Segmentation (Hindawi)

In this paper, Zhi-Hua Chen and Jung-Tae Kim present an efficient and effective method for hand gesture recognition. The hand region is detected through the background subtraction method. Then, the palm and fingers are split so as to recognize the fingers. After the fingers are recognized, the hand gesture can be classified through a simple rule classifier.

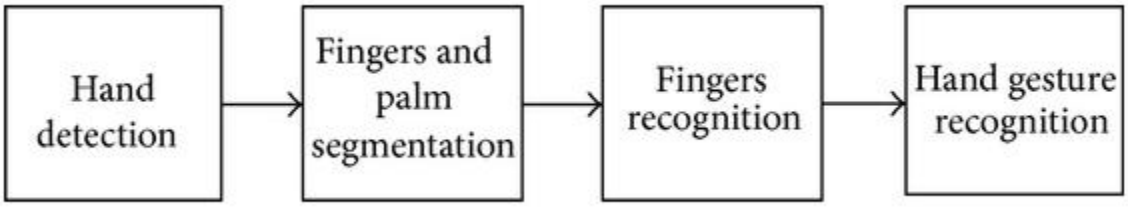


Fig 11: Base paper methodology

2.3 Tracking of dynamic gesture fingertips position in a video sequence

The field of research of this paper combines Human-Computer Interface, gesture recognition, and fingertip tracking. Most gesture recognition algorithms processing color images are unable to locate folded fingers hidden inside hand contour. With the use of hand landmarks detection and localization algorithm, processing directional images, the fingertips are tracked whether they are risen or folded inside the hand contour. The capabilities of the method, repeatability, and accuracy, are tested with the use of 3 gestures that are recorded on the USB camera. Fingertips are tracked in gestures presenting a linear movement of an open hand, finger folding into a fist, and clenched fist movement. In conclusion, a discussion of accuracy in application to HCI is presented.

2.4

CHAPTER 4 -EXPERIMENTAL OR MATERIALS AND METHODS; ALGORITHMS USED

3.1 Exploratory Data Analysis

Confusion Matrix

3.2 Dataset

The dataset is prepared by enabling a webcam. Labels should be binarized in a one-to-all fashion.Scikit-learn includes a number of regression and binary classification techniques. The so-called one-vs-all approach is a straightforward way to extend these algorithms to the multi-class classification problem.This basically entails learning one regressor or binary classifier per class throughout learning time. To do so, multi-class labels must be converted to binary labels (belong or do not belong to the class). The transform method in LabelBinarizer makes this operation simple.When it comes to prediction, one assigns the class for which the matching model provided the highest level of confidence. The inverse transform method in LabelBinarizer makes this simple.The dataset was mostly built utilising a live video feed to capture all of the signs from A to Z and 1 to 9 in jpeg format.Some experimental data has been acquired by influencing and changing the data.It accomplishes the study's goal of recognising Indian sign language from a video feed.

3.3 Proposed Methodology

CNN (Convolutional Neural Networks) is used in the suggested system. It's a four-layer modified CNN model that starts with three nodes.Modules like the torch were used to load, divide, and test the data during the training phase.The epochs were set to ten, and the device was examined to determine whether Cuda was available; if it wasn't, the CPU was used.The photo positions and labels were afterwards obtained from the

data.csv file. Adam and Cross-Entropy Loss are the optimizer and criterion used for the functions, respectively.Each predicted class probability is compared to the real class intended result, which is either 0 or 1, to determine a score/loss that penalises the probability based on how far it deviates from the actual expected value.Data loader, optimizer, criterion, model, and train data are used to produce a function called "fit" for training.The parameters data loader, optimizer, criterion, model, and valid data are supplied to another validation function called "validate."The validate loss and accuracy are calculated using this function.Learning rate = 0.001, batch size = 128, and epochs = 10 are specified as learning parameters. Within the period range, "fit" is called to provide accuracy and loss for training.The model is retained for future use after it has been trained.To open the web camera and then show the gesture, which is anticipated, and the word is displayed on the screen to receive the output and truly predict a hand sign.

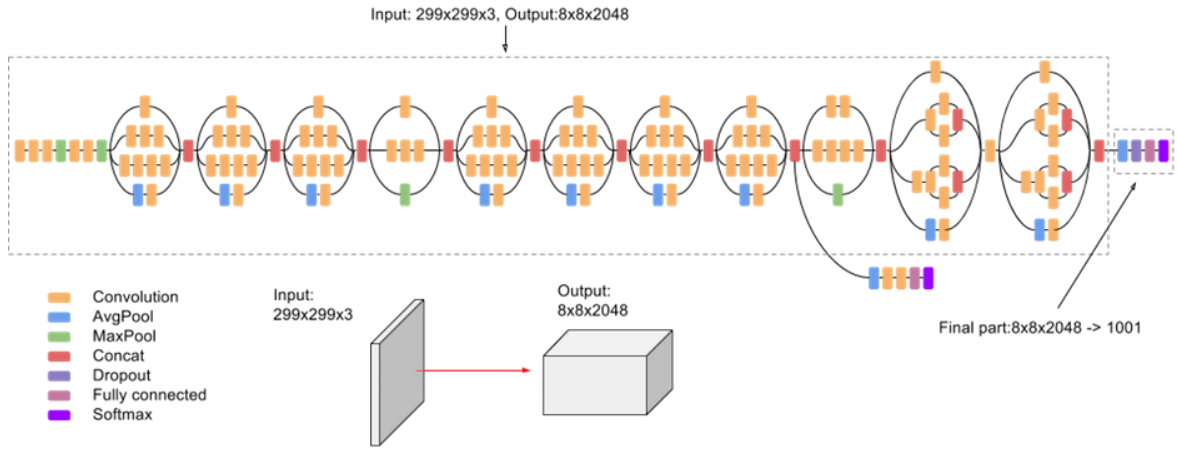


Figure 6: Inception V3 Architecture

3.3 Flowchart

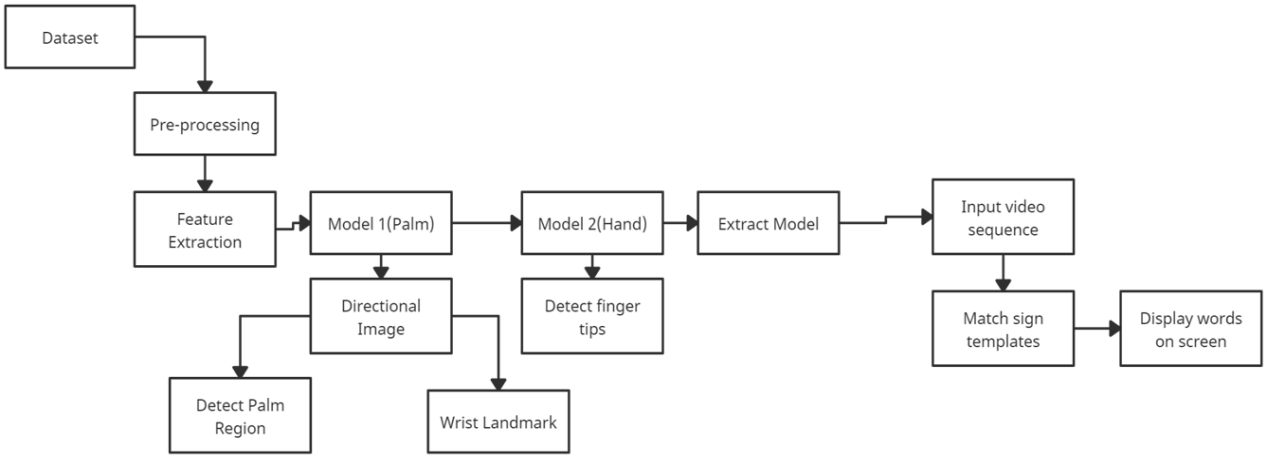


Figure 7 : Flowchart

5.1 Training and Validating

The data is trained using the CPU. As a result, CUDA is identified as zero (without GPU). The 42000 dataset is split into two parts: one for training and the other for validation.Training set: is a collection of samples used to learn how to fit the classifier's parameters [i.e., weights]. Validation set: A set of examples used to fine-tune a classifier's parameters [architecture, not weights], such as the number of hidden units in a neural network.

```
Computation device: cuda:0
Training on 35700 images
Validating on 6300 images
```

Figure 8 : Number of files

The CustomCNN function created is displayed below, which will be combined with the Inception Model to train the dataset.

```
CustomCNN(
  (conv1): Conv2d(3, 16, kernel_size=(5, 5), stride=(1, 1))
  (conv2): Conv2d(16, 32, kernel_size=(5, 5), stride=(1, 1))
  (conv3): Conv2d(32, 64, kernel_size=(3, 3), stride=(1, 1))
  (conv4): Conv2d(64, 128, kernel_size=(5, 5), stride=(1, 1))
  (fc1): Linear(in_features=128, out_features=256, bias=True)
  (fc2): Linear(in_features=256, out_features=35, bias=True)
  (pool): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
)
279,491 total parameters.
279,491 trainable parameters.
```

Figure 8: Custom CNN Architecture

The model is then fitted with two functions: one for training and the other for validating. This will produce the results, which will be compared to the target.

5.2 Results

Graph for train loss which shows the deviation of error loss while training. A loss function, also known as a cost function, is a function that transfers an event or the values of one or more variables onto a real number that intuitively represents some "cost" connected with the event in mathematical optimization and decision theory. This is the initial data epochs vs error loss. An epoch is a word used in machine learning that refers to the number of passes the machine learning algorithm has made across the full training dataset. At 0 epoch, loss calculated is approx. 0.005.

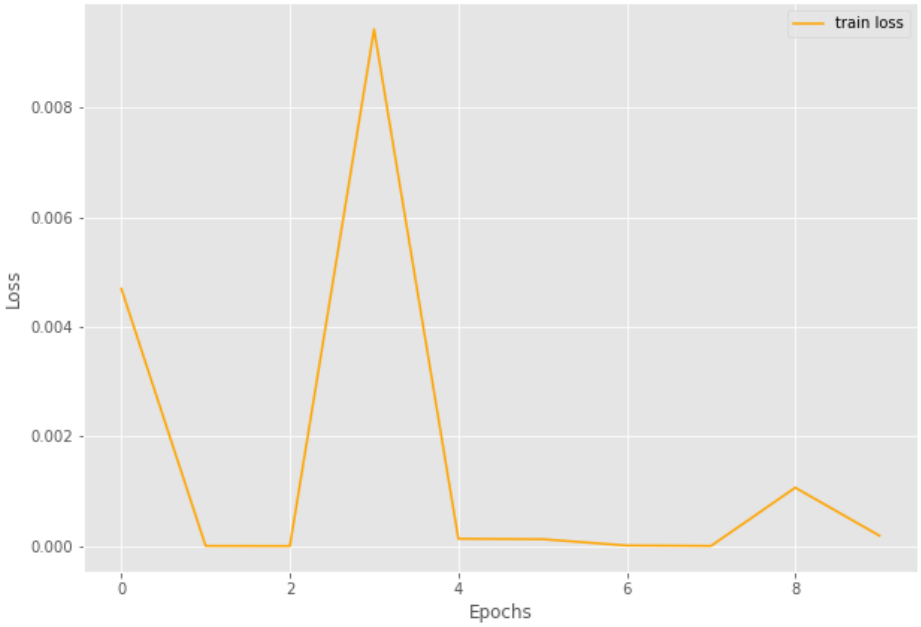


Figure 9. Train Loss

The graph below depicts the initial training accuracy, which indicates that the model was saved after a checkpoint. At 0 epoch, accuracy attained is approx 83%.As the training moves forward it changes to 98%.

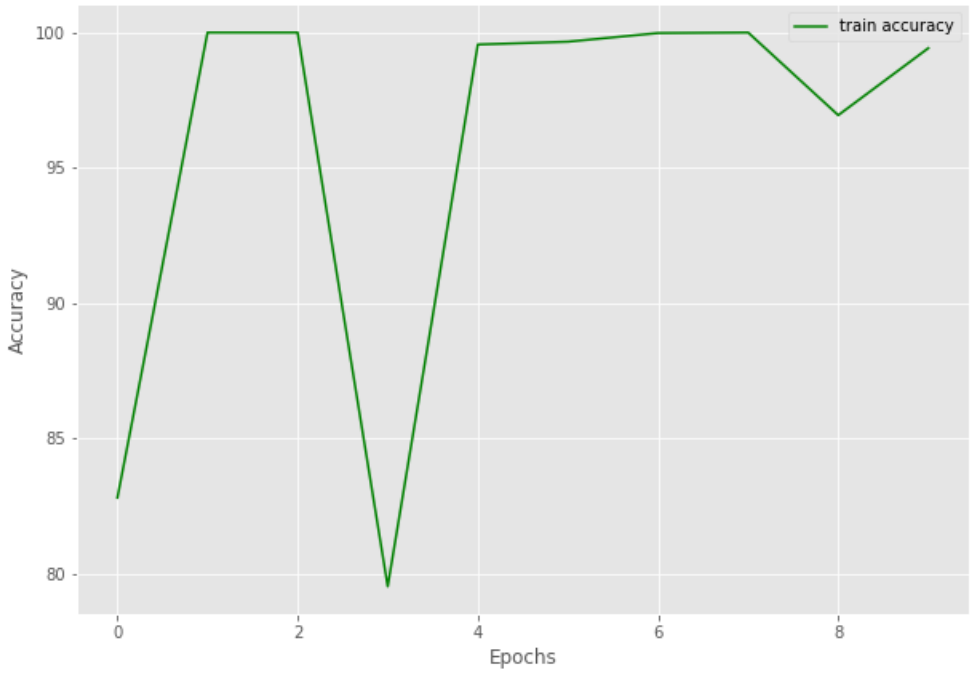
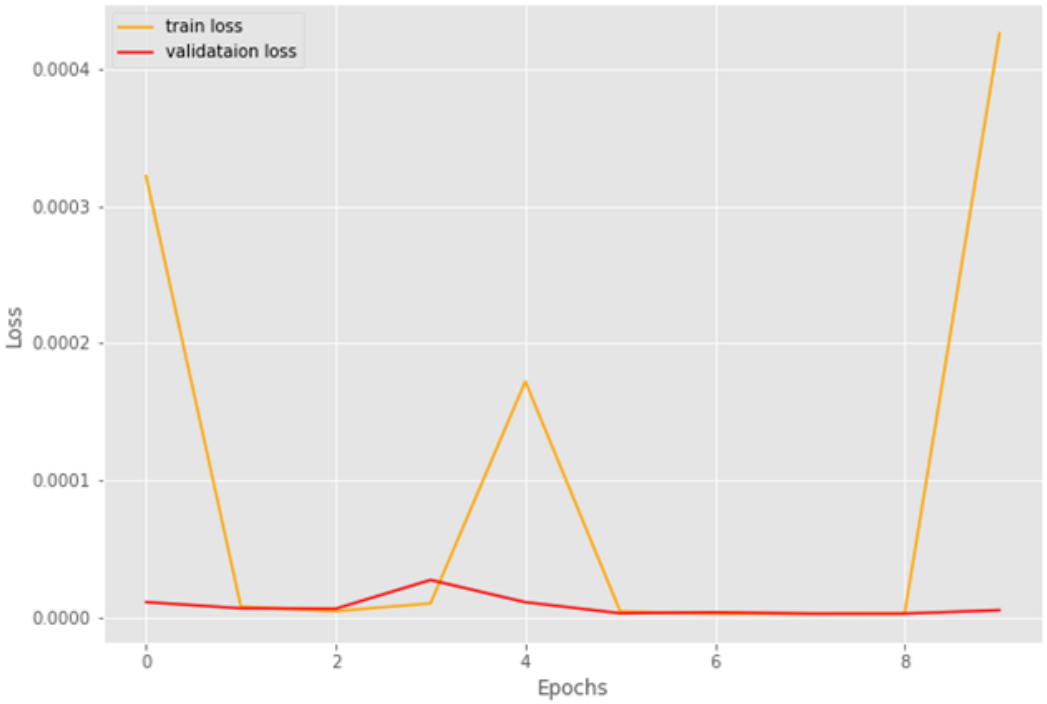
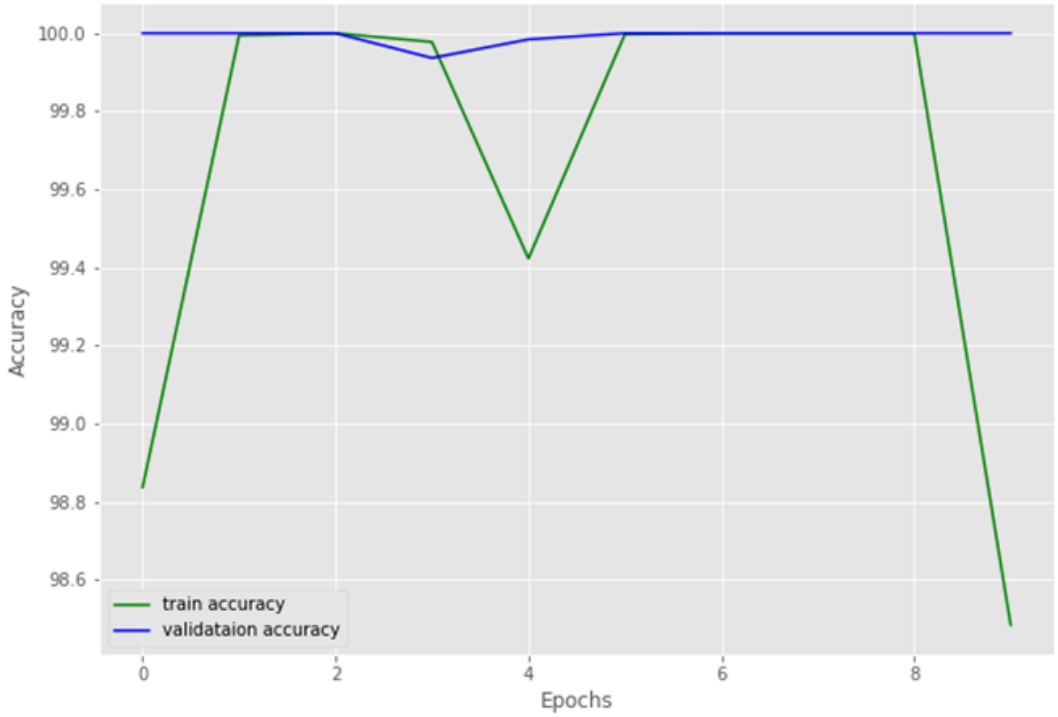
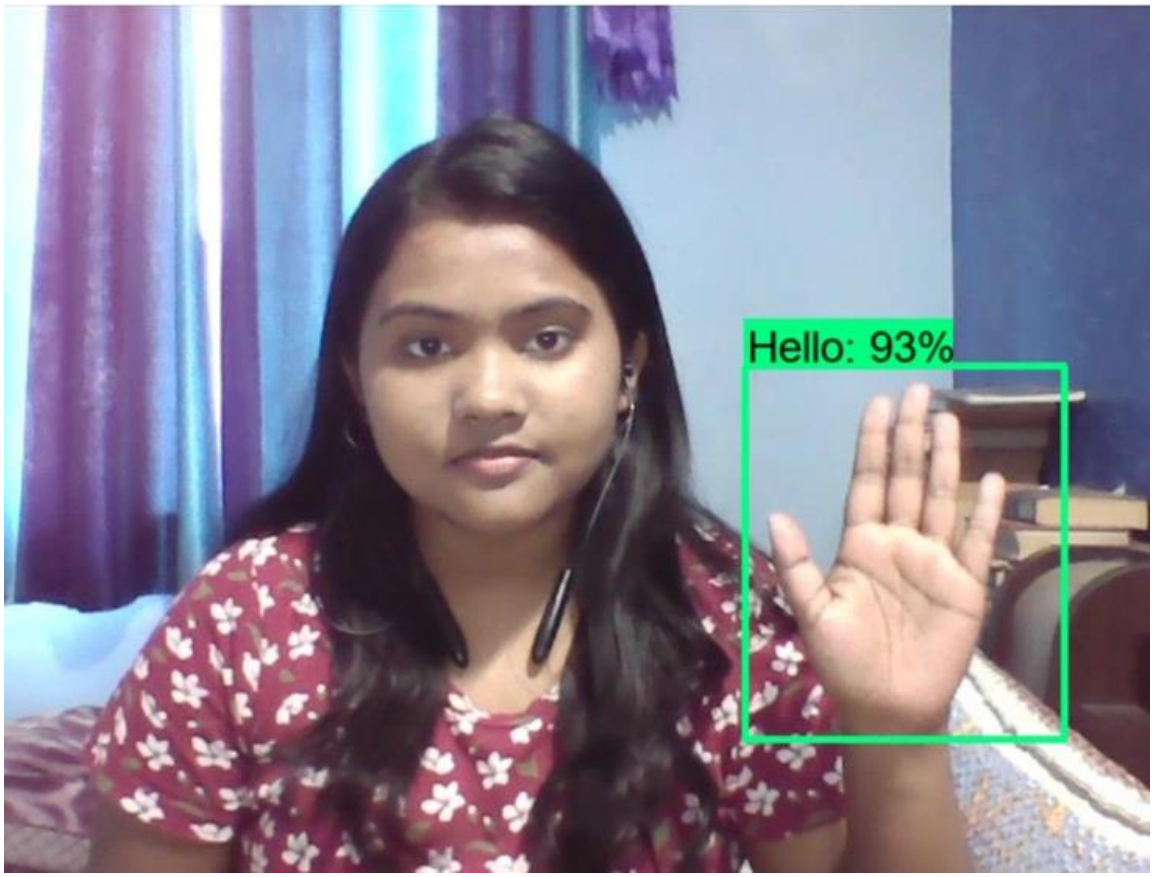


Figure 10. Training Accuracy


train loss
validataion loss
0.0004
0.0003
0.0002
0.0001
0.0000
Loss
0
2
4
6
8
Epochs


100.0
99.8
99.6
99.4
99.2
99.0
98.8
98.6
Accuracy
train accuracy
validataion accuracy
0
2
4
6
8
Epochs

:


Hello: 93%

CHAPTER 6 – CONCLUSION AND FUTURE WORK

6.1 Conclusion

Using a CustomCNN classifier, the output is a fully trained model with a 93 percent accuracy. With a total of 1200 samples, the visuals used were in Indian Sign Language. These photographs are organised into folders based on their meaning. The dataset was downloaded in jpeg format, then label binarized and pre-processed. We achieved a loss rate of roughly 0.10 after training the model. After that, the trained model was saved and reloaded. A script was used to enable the web camera. Finally, a hand gesture was demonstrated, which was effectively detected.

6.2 Future Work

In the future, this might be modified into an application where a person practicing sign language can transmit live video to the other end, which can be understood by the person watching. Basically, when the video is sequenced, it will capture the main gesture which classifies as the sign language, and then the recognized sign is classified. The database can be modified by adding more images to it. This will lead to making the system more accurate in detecting everything.

REFERENCES

NumPy, Pandas, MatplotLib: GeekForGeeks: geekforgeeks.org

Python3: www.python.org/download/releases/3.0/

Machine Learning: towardsdatascience.com

[1] Study of vision-based hand gesture recognition using Indian sign language in INTERNATIONAL JOURNAL ON SMART SENSING AND INTELLIGENT SYSTEMS VOL. 7, NO. 1, MARCH 2014.

[2] Real-time hand gesture recognition using finger segmentation in THE SCIENTIFIC WORLD JOURNAL (JAN 2014).

[3] Hand sign recognition from depth images with multi-scale density features for deaf-mute persons in International Conference on Computational Intelligence and Data Science (ICCIDS 2019).

[4] Indian sign language interpreter using image processing and machine learning in IOP Conference Series: Materials Science and Engineering 2020.

[5] Dynamic Hand Gesture Recognition: A Literature Review in IJERT 2012.

[6] Hand Gesture Recognition A Literature Review in IJSRD - International Journal for Scientific Research & Development| Vol. 8, Issue 2, 2020.

[7] Tracking of dynamic gesture fingertips position in a video sequence in Archives of Control Sciences Volume 30(LXVI), 2020.

[8] Visual tracking utilizing robust complementary learner and adaptive refiner in Science Direct,10th May 2017

[9] Tracking of dynamic gesture fingertips position in a video sequence in Archives of Control Sciences Volume 30(LXVI), 2020.

[10] Visual tracking utilizing robust complementary learner and adaptive refiner in Science Direct,10th May 2017.

[11] Sign Language Recognition Using Image Processing, Research Gate 2017.Development of the Recognition System of Japanese Sign Language Using 3D Image Sensor (HCI International 2013).