An Industry Oriented Mini Project (CS705PC)

on

SIGN LANGUAGE TRANSLATOR

Submitted in partial fulfilment of the requirements for the award of the degree of

Bachelor of Technology

in

Computer Science and Engineering

by

Mr.PREMALA VASANTH

(17261A0545)

Under the guidance of

Mr. S. K. IRFAN BABU

(Assistant Professor)

Department of Computer Science and Engineering

MAHATMA GANDHI INSTITUTE OF TECHNOLOGY

(Affiliated to Jawaharlal Nehru Technological University Hyderabad)

Kokapet(V), Gandipet(M), Hyderabad.

Telangana-500 075, India.

2020 - 2021

MAHATMA GANDHI INSTITUTE OF TECHNOLOGY

(Estd. in 1997 by Chaitanya Bharathi Educational Society)

(Affiliated to JNTU, Hyderabad; Accredited by NBA, AICTE-New Delhi)

Kokapet (village & gram panchayat), Rajendra Nagar (Mandal), RangaReddy(Dist.)

Chaitanya Bharathi P.O., Hyderabad-500 075.

CERTIFICATE

This is to certify that An Industry Oriented Mini Project (CS705PC) entitled **"SIGN LANGUAGE TRANSLATOR"**, being submitted by **PREMALA VASANTH** bearing Roll No: **17261A0545** in partial fulfilment of the requirements for the Award of the **Degree of Bachelor of Technology** in **Computer Science and Engineering** to **Jawaharlal Nehru University Hyderabad** is a record of bonafide work carried out by him under our guidance and supervision.

The results embodied in this project have not been submitted to any other University or Institute for the award of any degree or diploma.

Supervisor	Head of the Department
Mr. S. K. Irfan Babu	**Dr. C. R. K. Reddy**
Assistant Professor	Professor

External Examiner

DECLARATION

I hereby declare that the work reported in an Industry Oriented Mini Project (CS705PC), titled **“SIGN LANGUAGE TRANSLATOR”** is original and bonafide work carried out by me as a part of fulfilment for Bachelor of Technology in the Department of Computer Science and Engineering to Mahatma Gandhi Institute of Technology,Gandipet, Hyderabad (T.S) - 500075, is result of investigation carried out by under the guidance of Mr. S. K. Irfan Babu, Assistant Professor, Dept. Of CSE, MGIT.

No part of the work is copied from books/journals/internet and wherever the portion is taken, the same has been duly referred in the text. The report is based on the work done entirely by me and not copied from any other source.

PREMALA VASANTH

(17261A0545)

ACKNOWLEDGEMENT

I would like to express my sincere thanks to **Prof. K. Jaya Sankar, Professor & Principal MGIT**, for providing the work facilities in the college.

The satisfaction that accompanies the successful completion of any task would be incomplete without introducing the people who made it possible and whose constant guidance and encouragement crowns all efforts with success. They have been a guiding light and source of inspiration towards the completion of the project.

I would like to express my sincere gratitude and indebtedness to my Project Guide, **Mr. S. K. Irfan Babu, Assistant Professor, Dept. of CSE** who has supported me throughout the project with patience and knowledge.

I am also extremely thankful to my Project Coordinators,**Dr. C.R.K Reddy, Professor, Dept. of CSE and Ms.B. Prasanthi, Senior Assistant Professor, Dept. of CSE**, for their valuable suggestions and interest throughout the course of this project.

I am also thankful to **Dr. C.R.K Reddy, Professor & Head of Department**, Dept. of Computer Science and Engineering for providing excellent infrastructure and a conducive atmosphere for completing this project successfully.

Finally, I would like to take this opportunity to thank my family for their support throughout the work. I sincerely acknowledge and thank all those who gave directly or indirectly their support in completion of this project.

PREMALA VASANTH

(17261A0545)

INDEX

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

The term " dumb " is a word that used to address speech-impaired people. People affected by speech impairment can not communicate using hearing and speech, they completely rely on sign language for communicating with others. Sign language is used among everybody who is dumb, but they find a hard time communicating with people which are non-signers (normal people). Each ordinary person sees tunes in and responds to others. But there are some unfortunate people who don't have this significant god's gift. Such people, chiefly hard of hearing and unable to speak, rely upon correspondence by means of gestures-based communication with others. Be that as it may, correspondence with customary people is a significant disability for them since only one out of every odd run of the mill individuals grasp their gesture-based communication. So the requirement of a sign language translator is a must for speech-impaired people. There have been favorable progress in the field of gesture recognition with current advancements in deep learning. There is also been quite a significant advancement in computer vision which would enable us to easily track hand gestures. The system is to build up a framework that makes an interpretation of the communication via gestures into content that can be perused by anybody. This framework is called Sign Language Translator. It is the equivalent of real-time translation of hand gestures into equivalent English text.

This system takes hand gestures as input through video and translates it text which could be understood by a non-signer. There will be the use of CNN for the classification of hand gestures. This system will make communication speech impaired people less complex.

Keywords - CNN (Convolutional Neural Network),computer vision, gesture recognition, deep learning.

1. INTRODUCTION

Deaf and dumb is a term means a person who could neither hear nor speak or both. Many deaf and dumb persons are enrolled in a school system where learning takes place in a conventional way. They put effort to learn how to read and write using the traditional method of learning in addition to sign language which most of them understood well which most people without hearing impairment do not do [1]. However, despite these efforts, communication between people with hearing impairment and those without is extremely difficult especially in a classroom that joins these people together. These class of people will have difficulty in making friends with their core students and even teachers either within or outside the school environment because of the communication barrier [2].Anyway inside the setting of this exploration work, we might be worried about hand spelling part of the ASL, which is considered as an initial phase in such interpretation framework.

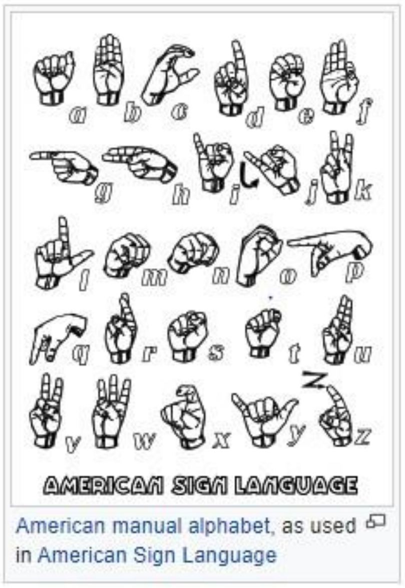


Figure 1.1: American Sign Language

Figure 1.1 describes Hand spelling alludes to the utilization of the different parts of a hand, to speak to letters in order, numerals, and uncommon characters. The latest and most updated hand spelling representation of the alphabet by the ASL [2] .

1.1 PROBLEM DEFINITION

Speech-impaired people around the world communicate with others using sign language. It is essential that the people that they are communicating with know sign language. But more often than not the other person doesn’t understand sign language [2]. Then the speech-impaired people are dependent on a sign language interpreter who would then translate the sign language to the people who don’t understand sign language. Speech-impaired people face huge problems conversing with non-signers. This is a pandemic problem not just around the world but in India also. The similar problem is being faced by signers in during educational purposes if the speech impaired hasn't understood a topic and has a doubt then he can't ask the doubts if the faculty doesn't understand sign language. So speech impaired people lots of problem in their day to day life if their interpreters isn't there [3].

1.2 EXISTING SYSTEM

The existing system for communication between signer and non-signer takes place through an interpreter. The Sign Language interpreter is well versed with Indo-Pakistani Sign Language Interpreter [1]. When the signer wants to convey a message to the non-signer he/she makes the appropriate sign language hand gestures to the interpreter and then the interpreter translates the hand gestures made by the signer into equivalent English. Similarly, the non-signer communicates to signer through the interpreter.

Disadvantages

- It reducing the communication gap between the signers and non-signers.
- This makes their informal and formal communication difficult.
- This process is quite tedious and time consuming and the signer may suffer if there is an absence of an interpreter.

1.3 PROPOSED SYSTEM

This proposed model/system could detect the hand gestures from a camera that is collecting as a set of frames, then preprocess the captured images and removed the background from each of the images using background-subtraction techniques. Using this set of frames the model recognizes the text of the corresponding hand gesture and displays it on the screen. Finally, the displayed text is translated into speech using API.

Advantages

- Architecture for a robust and accurate model for Hand gesture recognition..
- Doesn't require a priori information about the number of objects in the image.
- Works for both static and dynamic Hand gestures.

1.4 REQUIREMENTS SPECIFICATION

1.4.1 SOFTWARE REQUIREMENTS

1. Operating System: Windows 10
2. Software: PyCharm 2019.2.6 or Above

1.4.2 HARDWARE REQUIREMENTS

1. RAM: 8 GB
2. Hard Disk: 250 GB
3. Camera:Laptop Web Camera
4. CPU:Intel 5i Processor
5. GPU:2GB or Above(Recommended)
6. Input Devices: Keyboard, Mouse
7. Output Devices:Monitor,Speaker

2. LITERATURE SURVEY

1) **Sign Language Translator using Deep Learning (Uday Patil, Saraswati Nagtilak, Sunil Rai, Swaroop Patwari,Prajay Agarwal and Rahul Sharma)** [1], People affected by speech impairment can't communicate using hearing and speech, they rely on sign language for communication. Sign language is used among everybody who is speech impaired, but they find a hard time communicating with people which are non-signers. So requirement of a sign language interpreter is a must for speech impaired people.

2) **Sign Language and Gesture Recognition (Ashish Sah and A. Arul Prakash)** [2], There are some unfortunate people who doesn't have this significant gift. Such people, chiefly hard of hearing and unable to speak, they rely upon correspondence by means of gesture based communication to associate with others. This framework is called Sign Language Translator and Gesture Recognition.

3) **Portable Sign Language Translator for Emergency Response Teams (Mannava Vivek and Vitapu Gnanasagar)** [7], This system details the development of a system that uses image classification models to identify the hand signs made by speech disabled persons. It allows for easier communication between the dumb/deaf person and others. The system utilizes deep learning in a portable device that helps emergency personnel understand the sign language made by the speech-disabled person without having to wait for an interpreter.

4) **Sign Language Recognition Using Neural Network (Kaustubh Jadhav, Abhishek Jaiswal, Abbas Munshi and MayureshYerendekar)** [8], A practical sign language translator is an essential way for communication between the deaf community and the general public. So here we present the development and implementation of an American Sign Language (ASL) fingerspelling translator based on a convolutional neural network.

5) **Indian sign language interpreter using image processing and machine learning (Shubhendu Apoorv)** [11], Sign language plays an important part in the hearing/speech impaired person‟s life as it is the prime medium of communication for them. In this paper, the aim is to build a reliable communication interpretation program for interpreting Indian sign language and converting it to a readable output.

Table 2.1: Literature Survey

S.No.	Title of the Referred Paper,Journal Name, Year of Publication	Name of Authors	Technique /Methodology/Algorithm	Merits	Demerits
1	Sign Language Translator using Deep Learning, EasyChair Preprint, February 4, 2020	Uday Patil, Saraswati Nagtilak, and Rahul Sharma	OpenCV, YOLO Algorithm and VGGNet Architecture	Detect objects in up to 45 frames per a second.	It can recognize only one language i.e., English(ASL).
2	Sign Language Translator and Gesture Recognition, EasyChair Preprint, June 2, 2020	Ashish Sah and A. Arul Prakash	Keras or Tensorflow and CNN Architecture.	Robustness,Correct. And Classification rate of 99.5%.	It can recognize only one language i.e., English(ASL)..
3	Portable Sign Language Translator for Emergency Response Teams,International Journal of Scientific Engineering Trends,April 2020	Mannava Vivek and Vitapu Gnanasagar	Tensorflow Light Model, YOLO Algorithm and CNN Architecture	Hand Gesture and Classification Recognition using a CNN model.	It can recognize only one language i.e., English(ASL).
4	Sign Language Recognition Using Neural Network,K.C. College of Engineering ,June 2020	Kaustubh Jadhav, Abbas Munshi and MayureshYerendekar	OpenCV,CNN Architecture and YOLO Algorithm	Accurate model for hand gesture Recognition..	Sufficient light is required to recognize hand gestures.
5	Indian Sign Language Interpreter Using Image Processing and Machine Learning,IOP Publishing,August 2020	Shubhendu Apoorv	OpenCV, YOLO Algorithm and Machine Learning Algorithms	Learning, Robustness, and Non Linearity.	It can recognize only one language i.e., Indian Sign Language.

3. DESIGN METHODOLOGY OF SIGN LANGUAGE TRANSLATOR

3.1 PROPOSED SYSTEM FLOW CHART

The flow chart explains the steps occurring to accomplish the objectives of the project.

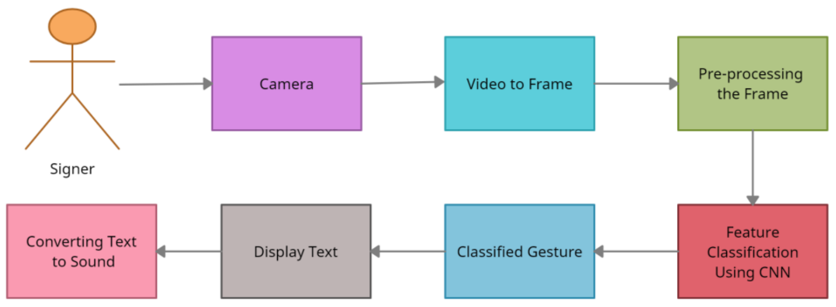


Figure 3.1: Flow Chart of the Proposed System

Figure 3.1 describes system architecture describes the steps in constructing a model that involves Gathering data, training the Deep Learning model to recognize ASL alphabets, and building the user interface. Initial processing shall be done using OpenCV [1] a popular library for computer vision developed by Intel. OpenCV [1] enables the conversion of video to frames. It also allows to downscale image and make it grayscale so that skin tone should have no impact on the final result as given by a neural network [1].

For gesture classification i.e. to identify which gesture has been made by the signers, we use CNN. Out of many CNN architectures Inception [12] shall be used. Using CNN, ASL alphabets shall be continuously classified until a word is formed and used for communication.

3.2 CNN ARCHITECTURE

3.2.1 GOOLENET/INCEPTION

Inception[1] is a convolutional neural network for assisting in image analysis and object detection, and got its start as a module for Googlenet. It is the of Google's Inception Convolutional Neural Network, originally introduced during the ImageNet Recognition Challenge. It is based on the original paper: "Rethinking the Inception Architecture for Computer Vision" by Szegedy [12]. The model itself is made up of symmetric and asymmetric building blocks, including convolutions, average pooling, max pooling, concats, dropouts, and fully connected layers. Loss is computed via Softmax.

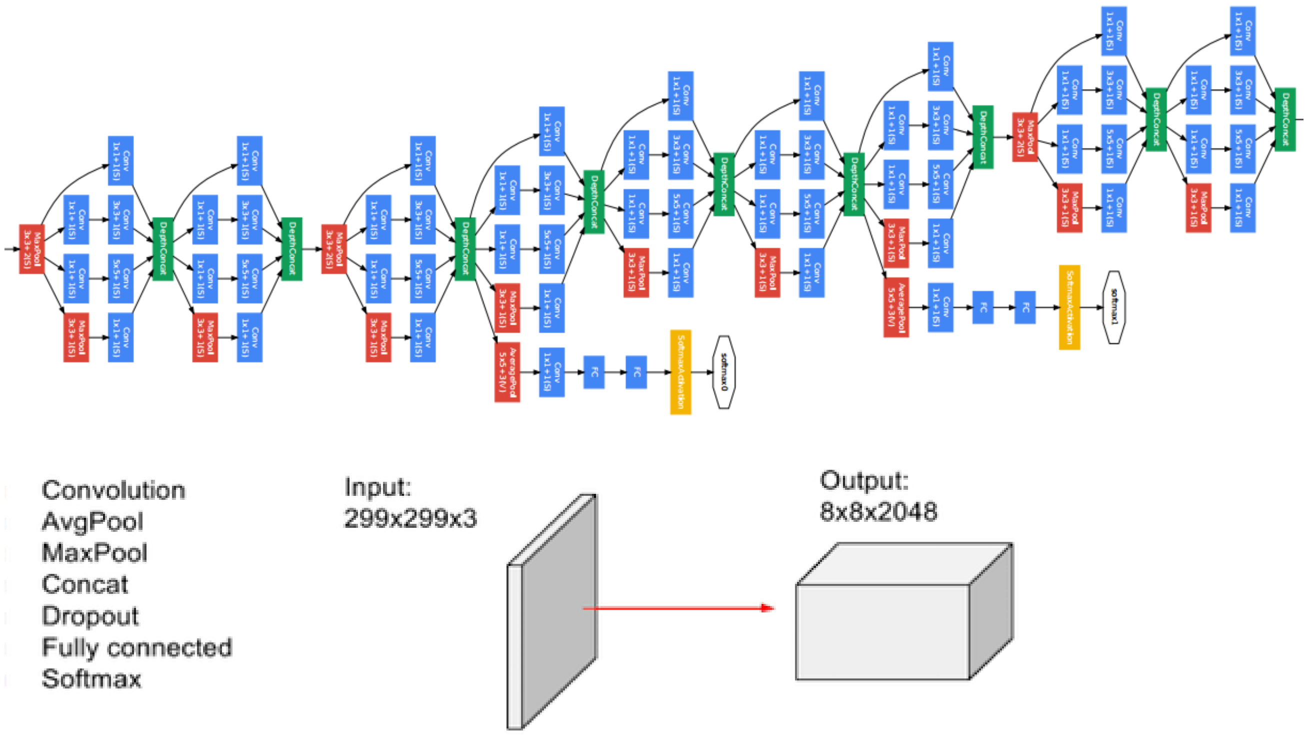


Figure 3.2: Inception Net architecture

Figure 3.2 describes Inception Network Architecture and its layers.

3.2.2 INCEPTION MODULE

Inception Modules are used in Convolutional Neural Networks to allow for more efficient computation and deeper Networks through a dimensionality reduction with stacked 1×1 convolutions. The modules were designed to solve the problem of computational expense, as well as overfitting, among other issues. The solution, in short, is to take multiple kernel filter sizes within the CNN, and rather than stacking them sequentially, ordering them to operate on the same level [12].

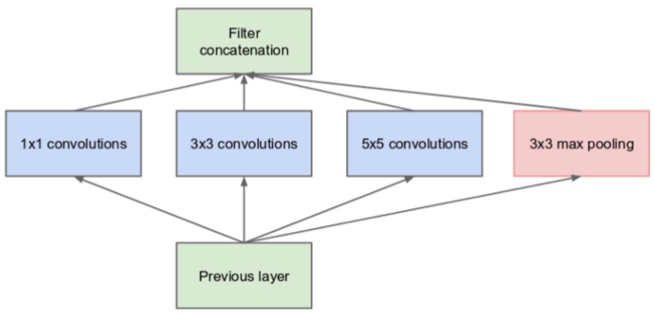


Figure 3.3: Inception Module,Naive Version

Figure 3.3 describes the naive version of the Inception module.Inception Modules are used in Convolutional Neural Networks to allow for more efficient computation and deeper Networks .

3.2.3 INCEPTION MODULE WITH DIMENSION REDUCTIONS

Inception Modules are incorporated into convolutional neural networks (CNNs) as a way of reducing computational expense. As a neural net deals with a vast array of images, with wide variation in the featured image content, also known as the salient parts, they need to be designed appropriately [12].

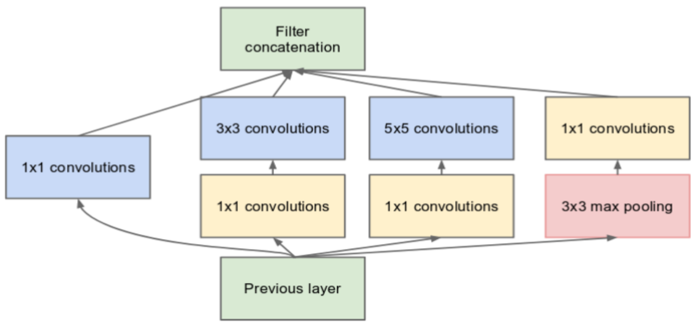


Figure 3.4: Inception Module With Dimension Reductions

Figure 3.4 describes Inception with dimensionality reduction, Inception Modules are used in CNN to allow for more efficient computation and deeper Networks through a dimensionality reduction.

3.2.4 FEATUES OF INCEPTION NETWORK

- **Input Layer**

It accepts color images as an input with the size 224 x 224 and 3 channels i.e.Red, Green, and Blue or Grayscale.

- **Convolution Layer**

The images pass through a stack of convolution layers where every convolution filter has a very small receptive field of 3 x 3 and stride of 1. Every kernel uses row and column padding so that the size of input as well as the output feature maps.

- **Max pooling**

It is performed over a max-pool window of different size with stride equals to 2,which means here max pool windows are non-overlapping windows.

- **Average pooling**

It involves calculating the average for each patch of the feature map. This means that each 2×2 square of the feature map is down sampled to the average value in the square.

- **Concatenate class Layer**

It concatenates a list of inputs. It takes as input a list of tensors, all of the same shape except for the concatenation axis, and returns a single tensor that is the concatenation of all inputs.

- **Dropout layer**

It randomly sets input units to 0 with a frequency of rate at each step during training time, which helps prevent overfitting.

- **Dense/fully connected layer**

A linear operation on the layer's input vector.

- **Softmax**

It converts a real vector to a vector of categorical probabilities.The elements of the output vector are in range (0, 1) and sum to 1.Each vector is handled independently. The softmax of each vector x is computed as exp(x) / tf.reduce_sum(exp(x)).

3.3 MODULE DESCRIPTION

3.3.1 DATASET

The Dataset available is of American Sign Language. The process of creating the dataset by capturing the images is a huge task. The data set is a collection of images of alphabets from the American Sign Language, separated in 44 folders which represent the various classes.The training data set contains 88,000 images which are 50x50 pixels [2]. There are 44 classes, of which 26 are for the letters A-Z,10 are for the digits 0-9 and 8 classes for Remember,Best of Luck,I Love You,I/Me,You etc..,.The test data set contains a mere 8800 images, to encourage the use of real-world test images and validation contains a mere 8800 images. We are keeping a minimum of 2400 plus images.

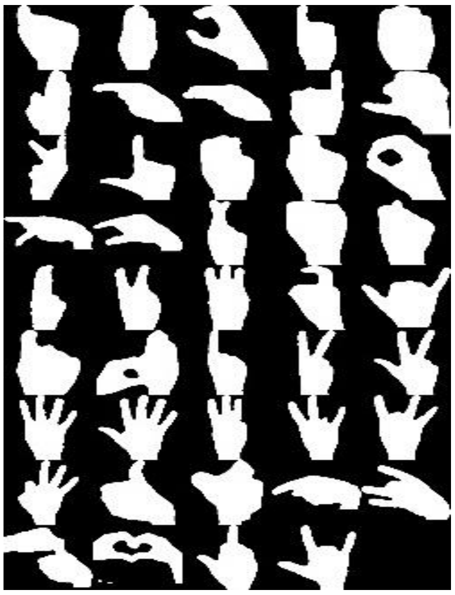


Figure 3.5: American Sign Language Classes

Figure 3.5 describes the different gestures/signs i,e., stored in the database.

3.3.2. IMPORTING MODULES

- **NumPy**

NumPy is the fundamental package for scientific computing with Python. As it is used to divide the given image from a construction site into n-dimensional objects such that it can be easily compared with the dataset.

- **Pandas**

Pandas is for data manipulation and analysis. The main purpose that we are using the pandas in the neural network because it analyses the given input data and it will recognize that the given thing is an object and it will further proceed.

- **Matplotlib**

Matplotlib is a Python 2D plotting library that is used to plot a given image in a number of small images and it will check with the dataset and it will produce publication-quality figures in a variety of hard copy formats and interactive environments across platforms.

- **OpenCV**

OpenCV was started at Intel in 1999 by Gary Bradsky and the first release came out in 2000. Vadim Pisarevsky joined Gary Bradsky to manage Intel's Russian software OpenCV team.

- **Pickle**

It is the process whereby a Python object hierarchy is converted into a byte stream, and "unpickling" is the inverse operation, whereby a byte stream is converted back into an object hierarchy.

- **glob**

The glob module finds all the pathnames matching a specified pattern according to the rules used by the Unix shell, although results are returned in arbitrary order. No tilde expansion is done, but *, ?, and character ranges expressed with [] will be correctly matched.

- **TensorFlow**

TensorFlow is a Python library for fast numerical computing created and released by Google. It is a foundation library that can be used to create Deep Learning models directly or by using wrapper libraries that simplify the process built on top of TensorFlow.

- **SQLite**

SQLite3 can be integrated with Python using sqlite3 module, which was written by Gerhard Haring. It provides an SQL interface compliant with the DB-API 2.0 specification described by PEP 249.

- **pyttsx**

Pyttsx is a text-to-speech conversion library in Python. Unlike alternative libraries, it works offline, and is compatible with both Python 2 and 3.

- **Thread**

The Python standard library provides threading , which contains most of the primitives you'll see in this article. Thread , in this module, nicely encapsulates threads, providing a clean interface to work with them.

3.4 DATASET PREPROCESSING

Hand Gestures are given as input in to the system in the video format. Then the video would be converted into individual frames. After that pre-processing would be carried out on the frames. In pre-processing the frames are converted from RGB to Greyscale [3] and the noise in the image is reduced and the size is reduced. The size is reduced that the computational power required will be less.

3.5 MODEL IMPLEMENTATION

Real time signal language to textual content and speech translation, specifically: 1. Reading man or woman signal gestures 2. Training the system learning model for image to textual content translation three. Forming words 4. Forming sentences 5. Forming entire content 6. Obtaining audio output [5].

3.5.1 IMAGE ACQUISITION

The gestures are captured through the web camera. This OpenCV video stream is used to capture the entire signing duration [5]. The frames are extracted from the stream and are processed as grayscale images with the dimension of 50*50.

3.5.2 HAND REGION SEGMENTATION & HAND DETECTION AND TRACKING

The captured images are scanned for hand gestures. This is a part of preprocessing before the image is fed to the model to obtain the prediction. The segments containing gestures are made more pronounced.

3.5.3 HAND POSTURE RECOGNITION

The preprocessed images are fed to the keras CNN model. The model that has already been trained generates the predicted label. All the gesture labels are assigned with a probability. The label with the highest probability is treated to be the predicted label [5].

3.5.4 DISPLAY AS TEXT & SPEECH

The model accumulates the recognized gesture to words.The recognized words are converted into the corresponding speech using the pyttsx3 library. The text to speech result is a simple work around but is an invaluable feature as it gives a feel of an actual verbal conversation.

- **Convolutional Neural Network for Detection**

CNN are a class of neural network that are highly useful in solving computer vision problems. They found inspiration from the actual perception of vision that takes place in the visual cortex of our brain. They make use of a filter/kernel to scan through the entire pixel values of the image and make computations by setting appropriate weights to enable detection of a specific feature.

- **The CNN Architecture functioning**

The CNN model for this project consists of 11 layers. There are 3 convolutional layers. The first convolutional layer, accepts an image with 50*50 size in the grayscale image. 16 filters of size 2*2 are used in this layer which results in the generation of an activation map of 49*49 for each filter which means the output is equivalent to 49*49*16. A rectifier linear unit (relu) layer is also added to eliminate any negative values on the map and replace it with 0 [3],[5].

A maxpooling layer is applied which reduces the activation to 25*25 by only considering maximum values in 2*2 regions of the map. This step increases the probability of detecting the desired feature. This is followed by a second convolutional layer [19]. This layer has 32 filters of size 3*3 which results in the generation of an activation map of 23*23 which means the output is equivalent to 23*23*32. A maxpooling layer further reduces the activation map to 8*8*32 by finding the maximum values in 3*3 regions of the map. 64 filters of size 5*5 reduce the input to an output of 4*4*64.

A maxpooling layer reduces the map to 1*1*64. The map is flattened to a 1d array of length 64. A dense layer expands the map to an array of 128 elements. A dropout layer drops out random map elements to reduce overfitting. In the end, a dense layer reduces the map to an array of 44 elements which represent the number of classes [5].

➢ **Recognition of Alphabets and number**

To discover bounding packing containers of various objects, as we used the Gaussian historical past subtraction which used a technique to version each history pixel with the resource of a mixture of K Gaussian set distributions. The historical past colorations are those stays longer are greater the static.

3.6 ALGORITHM

Algorithm Real time sign language conversion to text and Start.

Step-1: Set the hand histogram to adjust with the skin complexion and the lighting conditions.
Step-2: Apply data augmentation to the dataset to expand it and therefore reduce the overfitting.
Step-3: Split the dataset into train, test and validation data sets.
Step-4: Train the CNN model to fit the dataset.
Step-5: Generate the model report which includes the accuracy, error and the confusion matrix.
Step-6: Execute the prediction file – this file predicts individual gestures, cumulates them into words, displays the words as text, relays the voice output.

Stop

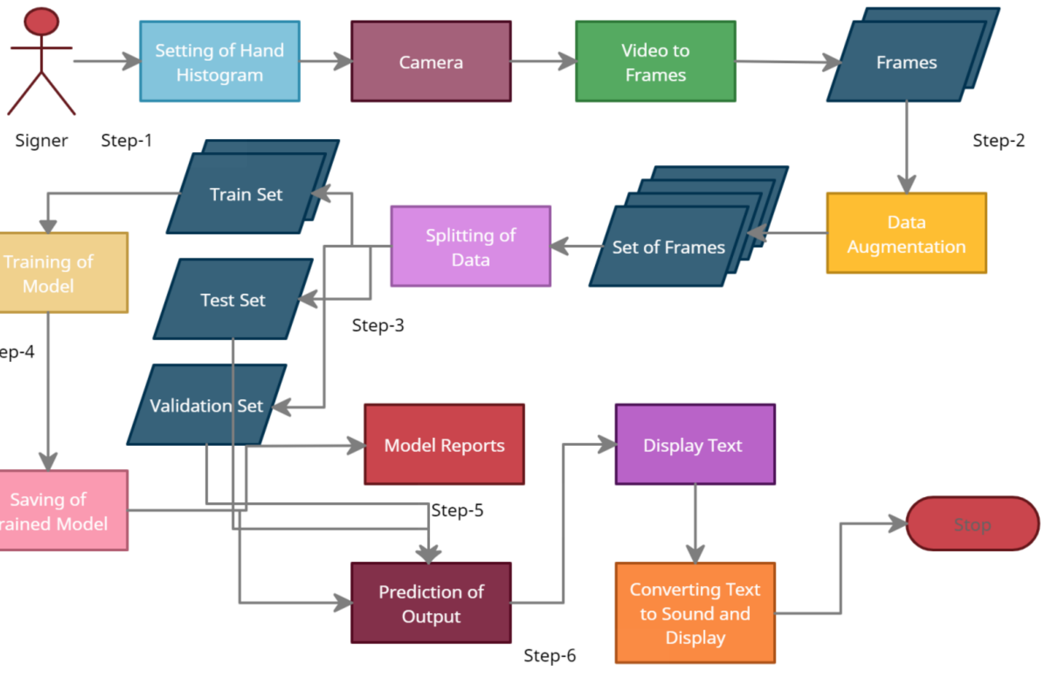


Figure 3.6 Diagrammatic Presentation of Algorithm

Figure 3.6 describes the step by step explanation of algorithm of proposed model.

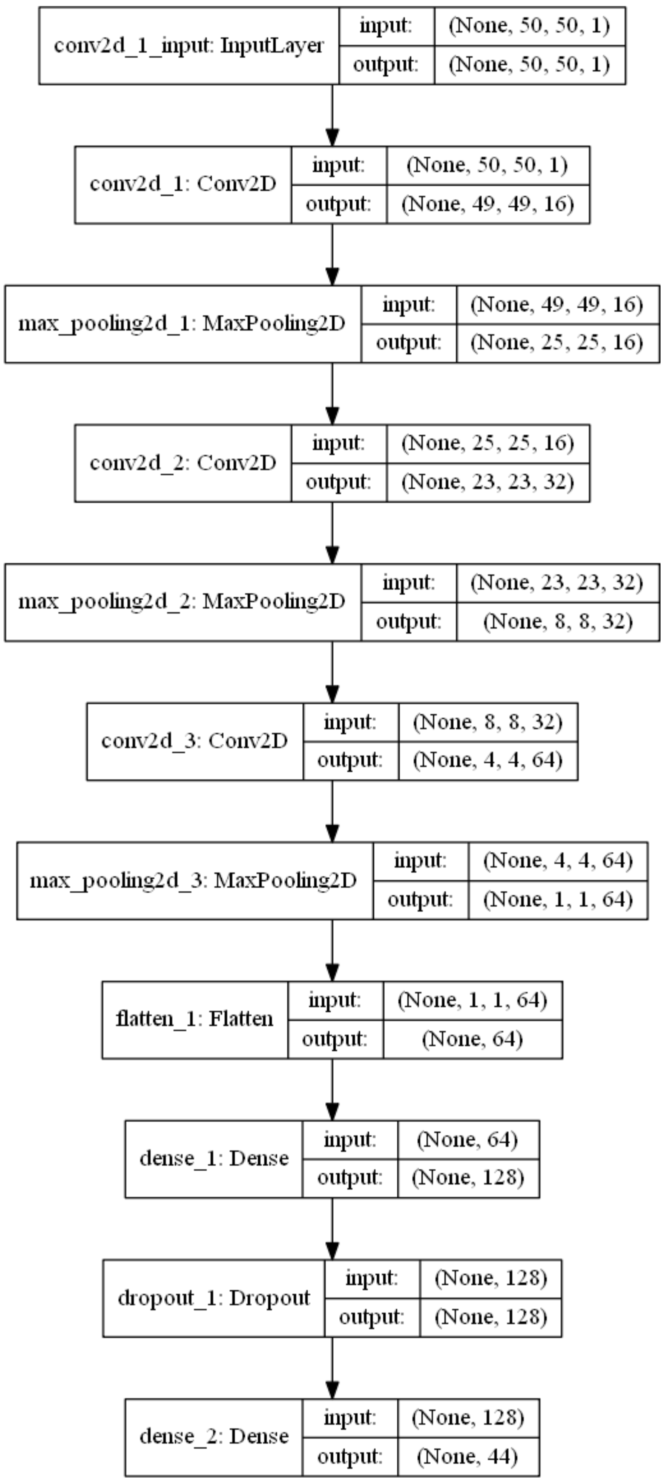

conv2d_1_input: InputLayer | input: | (None, 50, 50, 1) | output: | (None, 50, 50, 1)
conv2d_1: Conv2D | input: | (None, 50, 50, 1) | output: | (None, 49, 49, 16)
max_pooling2d_1: MaxPooling2D | input: | (None, 49, 49, 16) | output: | (None, 25, 25, 16)
conv2d_2: Conv2D | input: | (None, 25, 25, 16) | output: | (None, 23, 23, 32)
max_pooling2d_2: MaxPooling2D | input: | (None, 23, 23, 32) | output: | (None, 8, 8, 32)
conv2d_3: Conv2D | input: | (None, 8, 8, 32) | output: | (None, 4, 4, 64)
max_pooling2d_3: MaxPooling2D | input: | (None, 4, 4, 64) | output: | (None, 1, 1, 64)
flatten_1: Flatten | input: | (None, 1, 1, 64) | output: | (None, 64)
dense_1: Dense | input: | (None, 64) | output: | (None, 128)
dropout_1: Dropout | input: | (None, 128) | output: | (None, 128)
dense_2: Dense | input: | (None, 128) | output: | (None, 44)

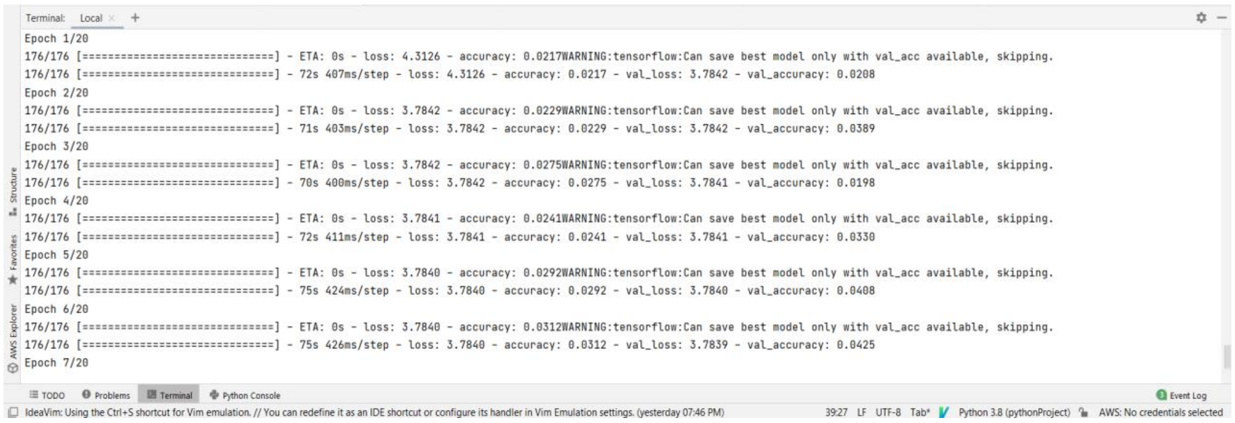
Figure 3.7: The CNN Architecture for the Project

Figure 3.7 describes the used CNN Architecture and its layers. This CNN are a class of neural network consists of 11 layers.

3.7 MODEL TRAINING

We are training model over 20 epochs.And with the batch size of 500 with the learning rate 0.38.We are using the model and validating using the validation data.Although we had a very large dataset to work with; 88,000 samples for each of 44 classes, after processing the images into numpy arrays, we found our personal computers could load a maximum of 2400 samples. The need to load small datasets actually led us to test the effect of increasing the data available to our models. On our preliminary model, which used strides of 2 on both layers we found the following relation between samples/class and model accuracy.

Training our initial models on less data led to the models quickly over fitting . This is likely due to the small amount of samples to train on leading to bad generalization and learning of the sample space. Increasing the size of our dataset to 200 samples/class led to better model results, with peak validation accuracy of 89.35% in epoch 17. However, taking a look at our loss function, we see that the validation loss is increasing, indicating over fitting of the model. After we applied filtering, enhancement, and ZCA [18] whitening to our dataset, the model performance increased drastically . The peak validation accuracy achieved is 95.93% in epoch 20.



```
Epoch 1/20
176/176 [==============================] - ETA: 0s - loss: 4.3126 - accuracy: 0.0217WARNING:tensorflow:Can save best model only with val_acc available, skipping.
176/176 [==============================] - 72s 407ms/step - loss: 4.3126 - accuracy: 0.0217 - val_loss: 3.7842 - val_accuracy: 0.0208
Epoch 2/20
176/176 [==============================] - ETA: 0s - loss: 3.7842 - accuracy: 0.0229WARNING:tensorflow:Can save best model only with val_acc available, skipping.
176/176 [==============================] - 71s 403ms/step - loss: 3.7842 - accuracy: 0.0229 - val_loss: 3.7842 - val_accuracy: 0.0389
Epoch 3/20
176/176 [==============================] - ETA: 0s - loss: 3.7842 - accuracy: 0.0275WARNING:tensorflow:Can save best model only with val_acc available, skipping.
176/176 [==============================] - 70s 400ms/step - loss: 3.7842 - accuracy: 0.0275 - val_loss: 3.7841 - val_accuracy: 0.0198
Epoch 4/20
176/176 [==============================] - ETA: 0s - loss: 3.7841 - accuracy: 0.0241WARNING:tensorflow:Can save best model only with val_acc available, skipping.
176/176 [==============================] - 72s 411ms/step - loss: 3.7841 - accuracy: 0.0241 - val_loss: 3.7841 - val_accuracy: 0.0330
Epoch 5/20
176/176 [==============================] - ETA: 0s - loss: 3.7840 - accuracy: 0.0292WARNING:tensorflow:Can save best model only with val_acc available, skipping.
176/176 [==============================] - 75s 424ms/step - loss: 3.7840 - accuracy: 0.0292 - val_loss: 3.7840 - val_accuracy: 0.0408
Epoch 6/20
176/176 [==============================] - ETA: 0s - loss: 3.7840 - accuracy: 0.0312WARNING:tensorflow:Can save best model only with val_acc available, skipping.
176/176 [==============================] - 75s 426ms/step - loss: 3.7840 - accuracy: 0.0312 - val_loss: 3.7839 - val_accuracy: 0.0425
Epoch 7/20
```

IdeaVim: Using the Ctrl+S shortcut for Vim emulation. // You can redefine it as an IDE shortcut or configure its handler in Vim Emulation settings. (yesterday 07:46 PM)

Figure 3.8: This Image Represents Training of Mode

Figure 3.8 describes the model i.e..,training model over 20 epochs.And with the batch size of 500 with the learning rate 0.38.

3.8 UML DIAGRAMS

Unified Modelling Language is a standard language for writing software blueprints. It can be used to visualize, specify, construct, and document the artifacts of a software-intensive system. Modelling is a proven and well-accepted engineering technique [19].

3.8.1 DATA FLOW DIAGRAM

Data Flow Diagram is a graphical representation of the flow of the data through an information system modelling its process aspects. Firstly, read the dataset and preprocess the dataset. Next, Train learning model over the dataset and test the model with the test dataset. Note down the accuracy given by that model and iterate the above process in order to get the best model which gives maximum accuracy.

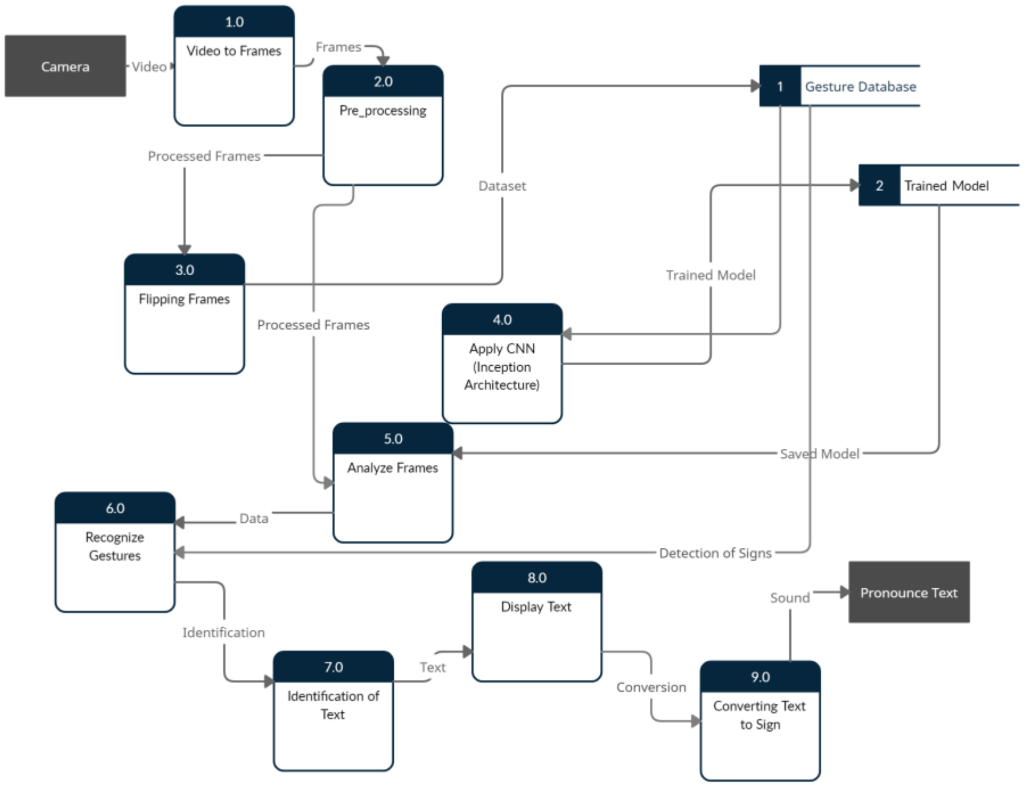


Figure 3.9: Data Flow Diagram

Figure 3.9 describes the data flow diagram of the proposed system.

3.8.2 USE - CASE DIAGRAM

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

The data set used should allow operations like cleaning the blurred image, updating, adding and deleting. The other actor is the user and system who deploys a model based on the maximum accuracy for an application.

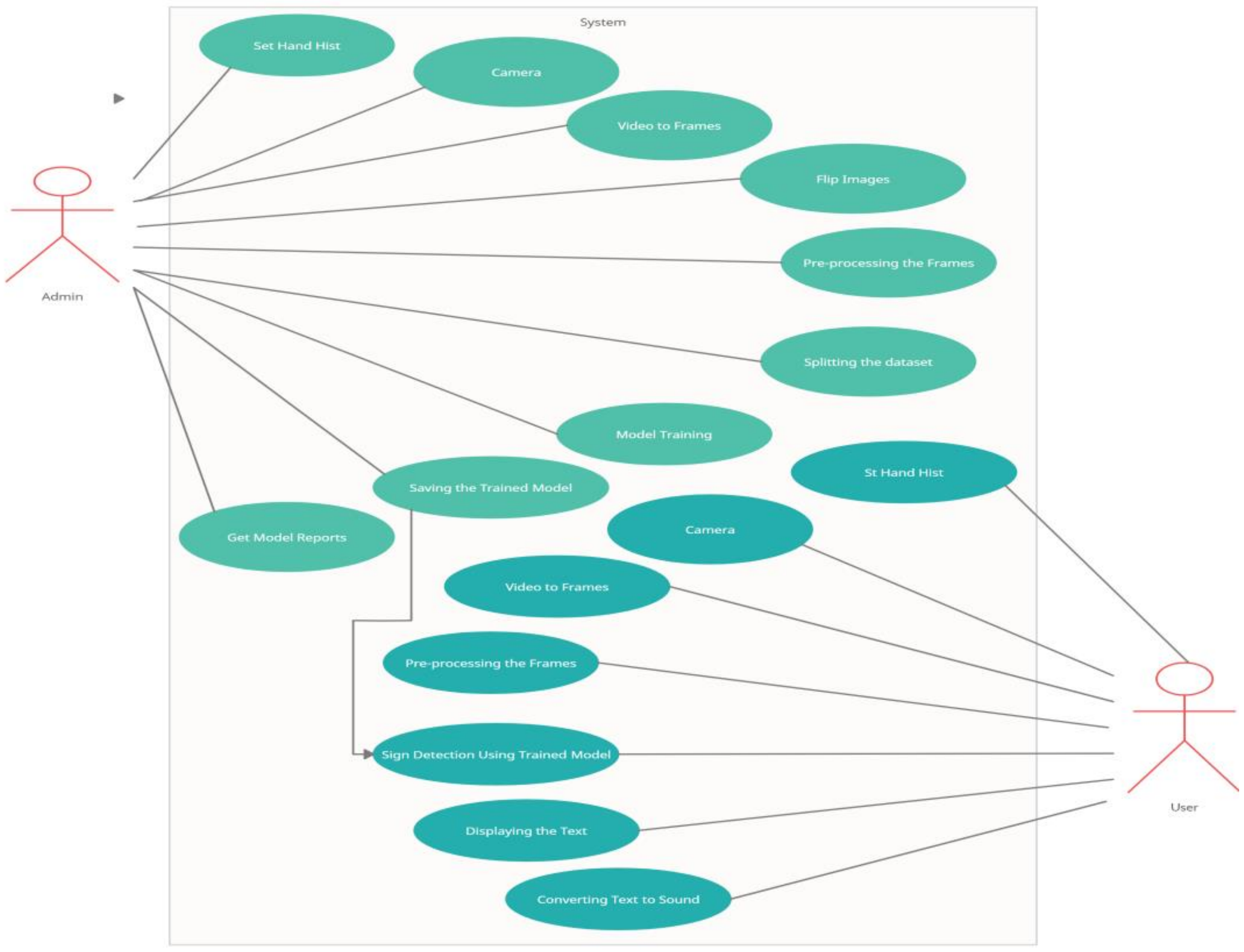


Figure 3.10: Use-Case Diagram

Figure 3.10 describes the use-case diagram of the proposed system and the role of the system administrator is to give input.

3.8.3 CLASS DIAGRAM

In software engineering, a class diagram in the Unified Modelling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations , and the relationships among the classes. Out of which User class and Designer class would provide the access to the respective person.

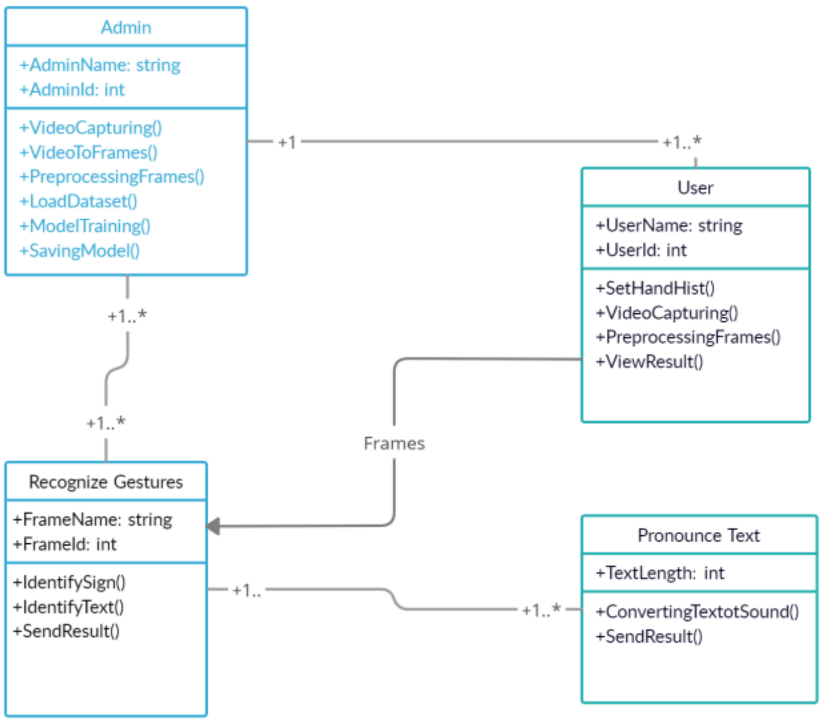


Figure 3.11: Class Diagram

Figure 3.11 describes the class diagram of the proposed system and there are four main classes to implement this algorithm.

3.8.4 SEQUENCE DIAGRAM

Sequence Diagram emphasizes the time ordering of messages. A sequence diagram shows a set of objects and the messages sent and received by those objects. Sequence diagrams are used to model the dynamic aspects of a system. If there are any noises in the data, clean the data using data cleaning operations such as cleaning the blurred parts and transform into clear image.

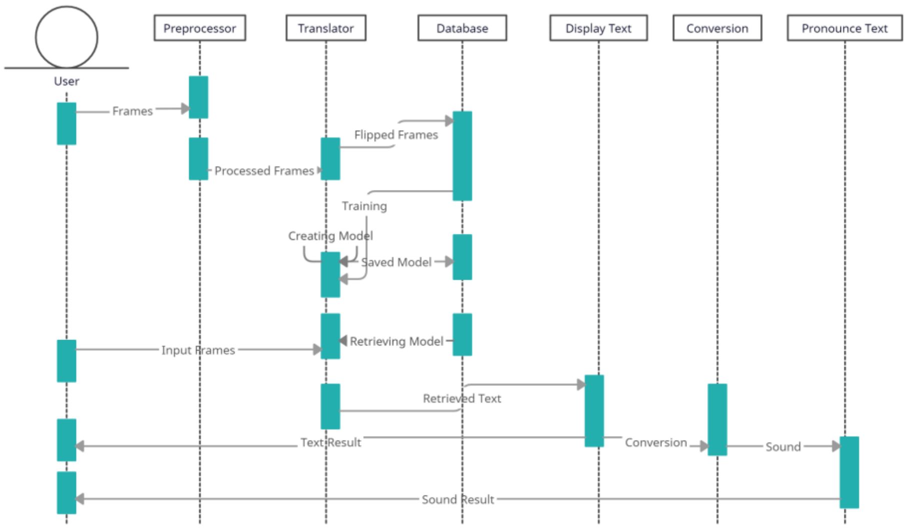


Figure 3.12: Sequence Diagram

Figure 3.12 describes the sequence diagram shows a set of objects.

3.8.5 ACTIVITY DIAGRAM

Activity diagram is another important behavioral diagram in UML diagram to describe dynamic aspects of the systems.Activity diagram is essentially an advanced version of the flow chart that is modeling the flow from one activity to another.

Activity diagram is used to show the steps of each method and also show the activity of the project. It can be used to represent the path of execution of a project. First we will load data,then annotate the data ,then split the data to test and train sets, then try to build a model on that so that it can predict the defect on the metallic surface.

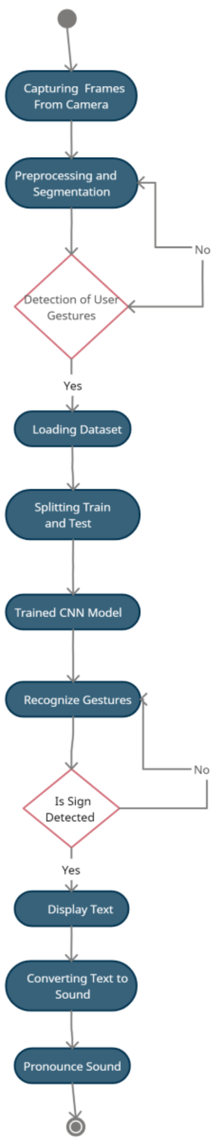


Figure 3.13: Activity Diagram

Figure 3.13 describes activity diagram of flow of sequence of actions in the proposed system.

4.TESTING AND RESULTS

Testing is a very important module in the software development to verify, validate and provide quality and service for different components of software.

4.1 MODEL TRAINING REPORT

```
(8800, 44)
Model: "sequential_2"
_________________________________________________________________
Layer (type) Output Shape Param #
=================================================================
conv2d_4 (Conv2D) (None, 49, 49, 16) 80
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 25, 25, 16) 0
_________________________________________________________________
conv2d_5 (Conv2D) (None, 23, 23, 32) 4640
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 8, 8, 32) 0
_________________________________________________________________
conv2d_6 (Conv2D) (None, 4, 4, 64) 51264
_________________________________________________________________
max_pooling2d_6 (MaxPooling2 (None, 1, 1, 64) 0
_________________________________________________________________
flatten_2 (Flatten) (None, 64) 0
_________________________________________________________________
dense_3 (Dense) (None, 128) 8320
_________________________________________________________________
dropout_2 (Dropout) (None, 128) 0
_________________________________________________________________
dense_4 (Dense) (None, 44) 5676
=================================================================
Total params: 69,980 Trainable params: 69,980
Non-trainable params: 0
_________________________________________________________________

Train on 88000 samples, validate on 8800 samples
Epoch 1/20
88000/88000 [==============================] - 129s 1ms/step - loss: 3.0812 - accuracy:
0.2897 - val_loss: 0.4699 - val_accuracy: 0.9053
Epoch 2/20
88000/88000 [==============================] - 141s 2ms/step - loss: 0.3587 - accuracy:
0.8929 - val_loss: 0.0470 - val_accuracy: 0.9875
Epoch 3/20
88000/88000 [==============================] - 154s 2ms/step - loss: 0.1123 - accuracy:
0.9655 - val_loss: 0.0185 - val_accuracy: 0.9953
Epoch 4/20
88000/88000 [==============================] - 131s 1ms/step - loss: 0.0636 - accuracy:
0.9810 - val_loss: 0.0103 - val_accuracy: 0.9974
Epoch 5/20
88000/88000 [==============================] - 128s 1ms/step - loss: 0.0454 - accuracy:
```

Continuation……..
After the 19th Epoch

```
Epoch 20/20
88000/88000 [==============================] - 127s 1ms/step - loss: 0.0060 - accuracy:
0.9983 - val_loss: 0.0011 - val_accuracy: 0.9998
CNN Error: 0.05%
```

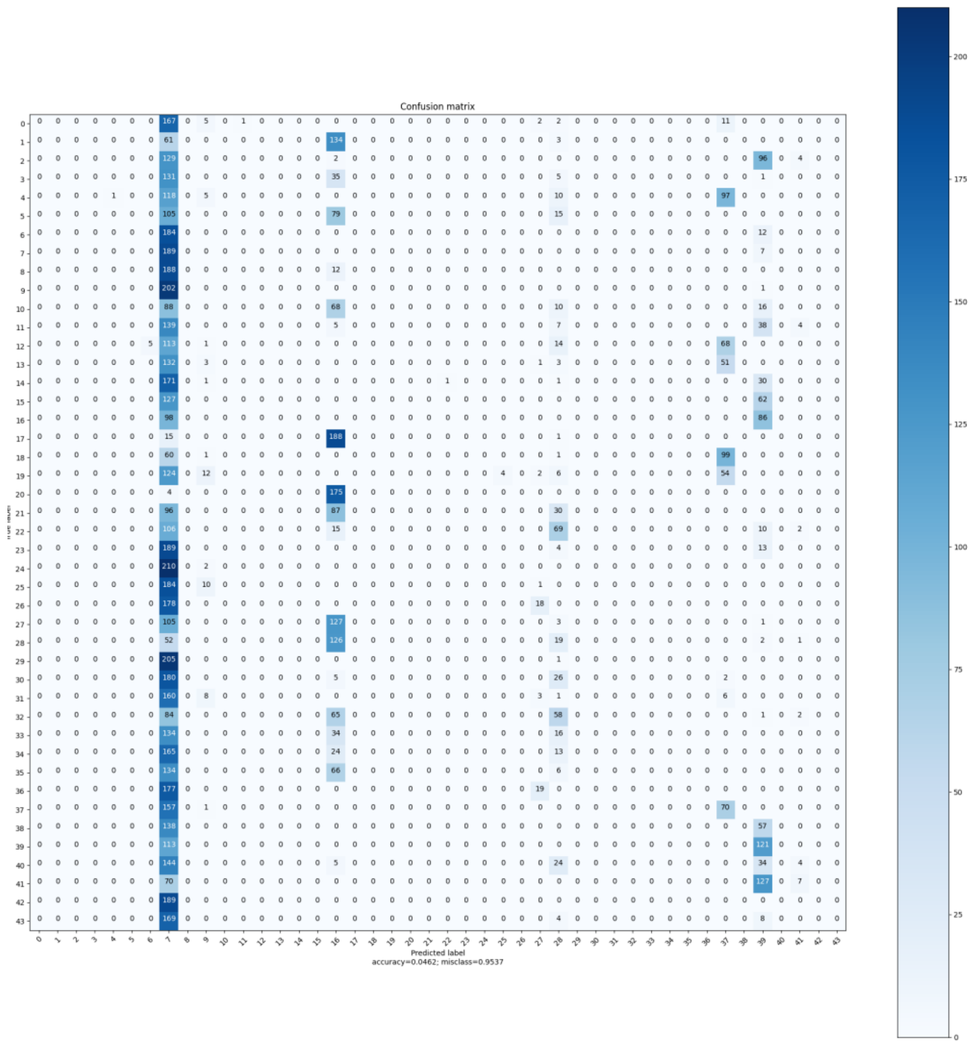


Figure 4.1: Classification Reports about the Model

Figure 4.1 describes metrics to evaluate the trained model and classification reports.

4.2 CLASSIFICATION REPORT

```
           precision    recall  f1-score   support

     0       0.99      0.99      0.99       188
     1       1.00      1.00      1.00       198
```

```
           2       1.00      1.00      1.00       231
           3       1.00      1.00      1.00       172
           4       1.00      1.00      1.00       231
           5       1.00      1.00      1.00       199
           6       1.00      1.00      1.00       196
           7       1.00      1.00      1.00       196
           8       1.00      0.99      1.00       200
           9       1.00      1.00      1.00       203
          10       1.00      1.00      1.00       182
          11       1.00      1.00      1.00       193
          12       1.00      1.00      1.00       201
          13       1.00      1.00      1.00       190
          14       1.00      1.00      1.00       204
          15       1.00      1.00      1.00       189
          16       1.00      0.99      1.00       184
          17       1.00      1.00      1.00       204
          18       0.99      1.00      1.00       161
          19       1.00      1.00      1.00       202
          20       1.00      1.00      1.00       179
          21       1.00      1.00      1.00       213
          22       1.00      1.00      1.00       202
          23       1.00      1.00      1.00       206
          24       1.00      1.00      1.00       212
          25       1.00      1.00      1.00       195
          26       1.00      1.00      1.00       196
          27       1.00      1.00      1.00       236
          28       1.00      1.00      1.00       200
          29       1.00      1.00      1.00       206
          30       1.00      1.00      1.00       213
          31       1.00      1.00      1.00       178
          32       1.00      1.00      1.00       210
          33       1.00      1.00      1.00       184
          34       1.00      1.00      1.00       202
          35       1.00      1.00      1.00       206
          36       1.00      1.00      1.00       196
          37       1.00      1.00      1.00       228
          38       1.00      1.00      1.00       195
          39       1.00      1.00      1.00       234
          40       1.00      1.00      1.00       211
          41       1.00      1.00      1.00       204
          42       1.00      1.00      1.00       189
          43       1.00      1.00      1.00       181

    accuracy                           1.00      8800
   macro avg       1.00      1.00      1.00      8800
weighted avg       1.00      1.00      1.00      8800
```

4.2.1 FORMULAS

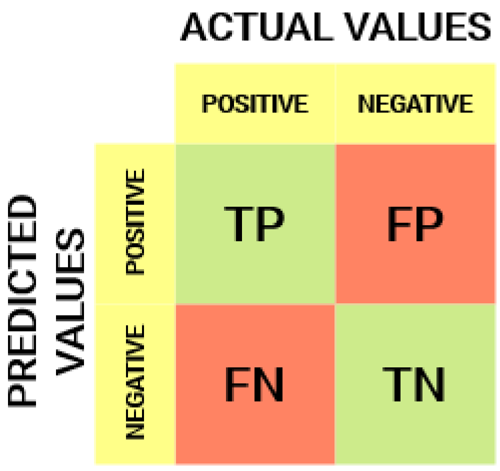


Figure 4.2 Confusion Matrix

Figure 4.2 describes the structure of confusion matrix.

Precision = TruePositives / (TruePositives + FalsePositives)

Recall = TruePositives / (TruePositives + FalseNegatives)

F1 Score = (2 * Precision * Recall) / (Precision + Recall)

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}}$$

- **Macro Average**

Average=macro says the function to compute f1 for each label, and returns the average without onsidering the proportion for each label in the dataset.

- **Weighted Average**

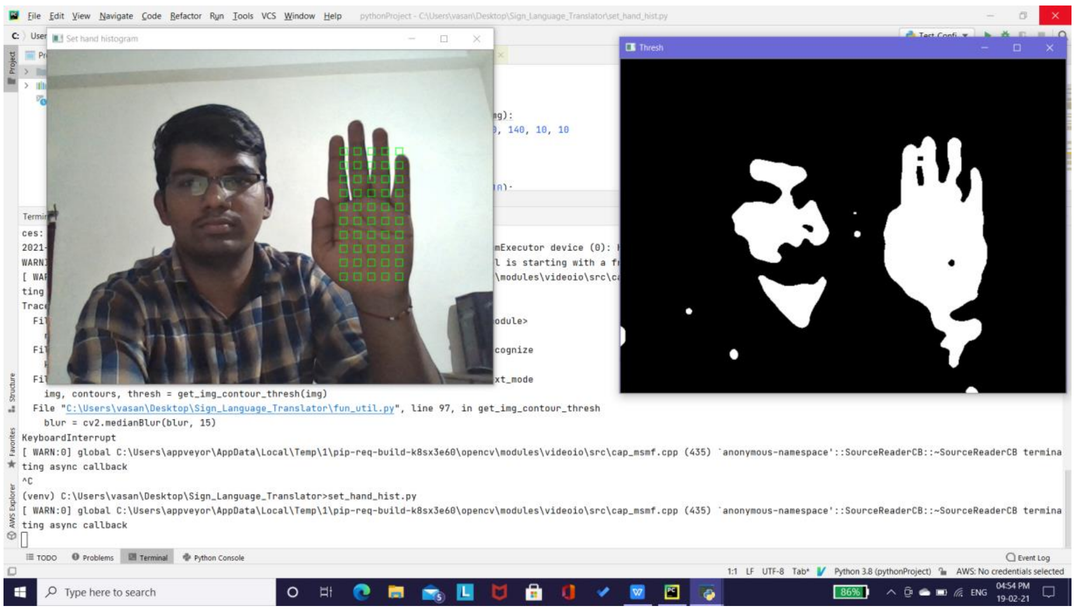
Average=weighted says the function to compute f1 for each label, and returns the average considering the proportion for each label in the dataset.

4.3 RESULTS

4.3.1 SETTING HISTOGRAM

First set your hand histogram. You do not need to do it again if you have already done it. But you do need to do it if the lighting conditions change. To do so type the command given below and follow the instructions below.

- ➢ A windows "Set hand histogram" will appear."Set hand histogram" will have 50 squares (5x10).Put your hand in those squares. Make sure your hand covers all the squares.
- ➢ Press 'c'. 1 other window will appear "Thresh".
- ➢ On pressing 'c' only white patches corresponding to the parts of the image which has your skin color should appear on the "Thresh" window.
- ➢ Make sure all the squares are covered by your hand.In case you are not successful then move your hand a little bit and press 'c' again.
- ➢ After you get a good histogram press 's' to save the histogram.


File Edit View Navigate Code Refactor Run Tools VCS Window Help
pythonProject - C:\Users\vasan\Desktop\Sign_Language_Translator\set_hand_hist.py
Set hand histogram
Thresh
TODO
Problems
Terminal
Python Console
Event Log
1:1 LF UTF-8 Tab* Python 3.8 (pythonProject) AWS: No credentials selected
Type here to search
86%
ENG
04:54 PM
19-02-21

```
  img, contours, thresh = get_img_contour_thresh(img)
File "C:\Users\vasan\Desktop\Sign_Language_Translator\fun_util.py", line 97, in get_img_contour_thresh
  blur = cv2.medianBlur(blur, 15)
KeyboardInterrupt
[ WARN:0] global C:\Users\appveyor\AppData\Local\Temp\1\pip-req-build-k8sx3e60\opencv\modules\videoio\src\cap_msmf.cpp (435) `anonymous-namespace'::SourceReaderCB::~SourceReaderCB termina
ting async callback
^C
(venv) C:\Users\vasan\Desktop\Sign_Language_Translator>set_hand_hist.py
[ WARN:0] global C:\Users\appveyor\AppData\Local\Temp\1\pip-req-build-k8sx3e60\opencv\modules\videoio\src\cap_msmf.cpp (435) `anonymous-namespace'::SourceReaderCB::~SourceReaderCB termina
ting async callback
```

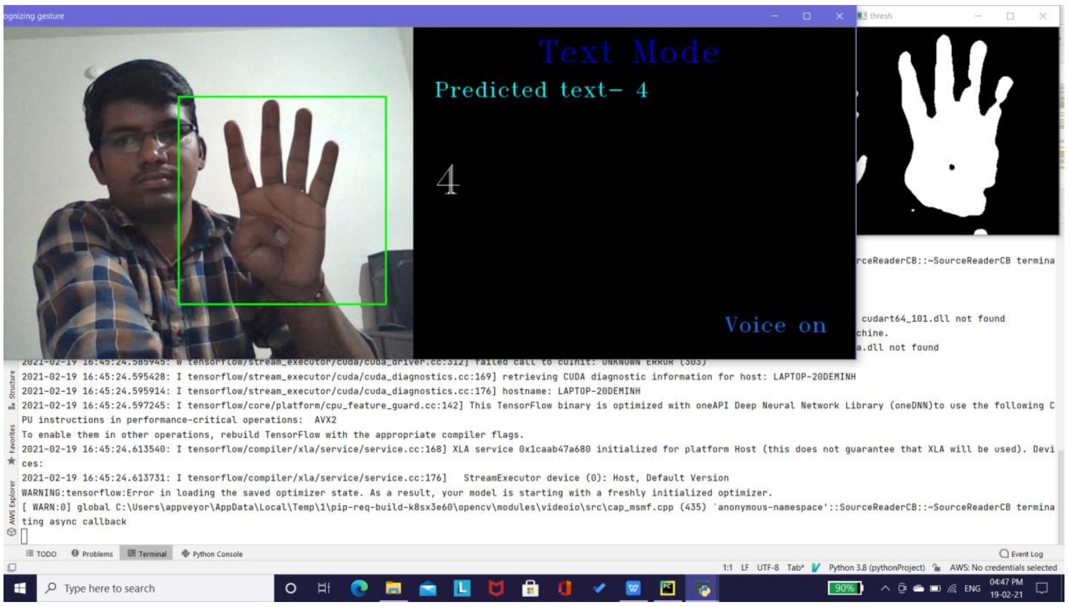
Figure 4.3: Setting of Hand Histogram

Figure 4.3 describes the setting of hand histogram.

4.3.2 SIGN RECOGNITION AND TRANSLATION

Text Mode

- ➢ In text mode you can create your own words using finger spellings or use the predefined gestures.
- ➢ The text on screen will be converted to speech on removing your hand from the green box
- ➢ Make sure you keep the same gesture on the green box for 15 frames or else the gesture will not be converted to text.

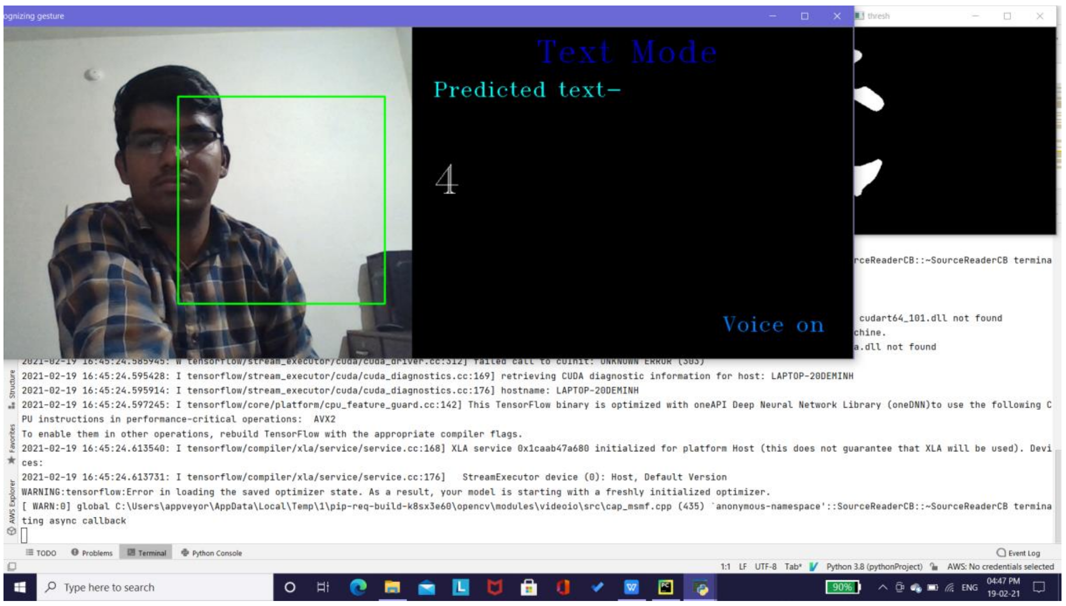


```
rceReaderCB::~SourceReaderCB termina

 cudart64_101.dll not found
chine.
a.dll not found
2021-02-19 16:45:24.585945: W tensorflow/stream_executor/cuda/cuda_driver.cc:312] failed call to cuInit: UNKNOWN ERROR (303)
2021-02-19 16:45:24.595428: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:169] retrieving CUDA diagnostic information for host: LAPTOP-20DEMINH
2021-02-19 16:45:24.595914: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:176] hostname: LAPTOP-20DEMINH
2021-02-19 16:45:24.597245: I tensorflow/core/platform/cpu_feature_guard.cc:142] This TensorFlow binary is optimized with oneAPI Deep Neural Network Library (oneDNN)to use the following C
PU instructions in performance-critical operations:  AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
2021-02-19 16:45:24.613540: I tensorflow/compiler/xla/service/service.cc:168] XLA service 0x1caab47a680 initialized for platform Host (this does not guarantee that XLA will be used). Devi
ces:
2021-02-19 16:45:24.613731: I tensorflow/compiler/xla/service/service.cc:176]   StreamExecutor device (0): Host, Default Version
WARNING:tensorflow:Error in loading the saved optimizer state. As a result, your model is starting with a freshly initialized optimizer.
[ WARN:0] global C:\Users\appveyor\AppData\Local\Temp\1\pip-req-build-k8sx3e60\opencv\modules\videoio\src\cap_msmf.cpp (435) `anonymous-namespace'::SourceReaderCB::~SourceReaderCB termina
ting async callback
```

Figure 4.4: Displaying Text

Figure 4.4 describes the gesture recognition and displays its corresponding text.



```
rceReaderCB::~SourceReaderCB termina

 cudart64_101.dll not found
chine.
a.dll not found
2021-02-19 16:45:24.585945: W tensorflow/stream_executor/cuda/cuda_driver.cc:312] failed call to cuInit: UNKNOWN ERROR (303)
2021-02-19 16:45:24.595428: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:169] retrieving CUDA diagnostic information for host: LAPTOP-20DEMINH
2021-02-19 16:45:24.595914: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:176] hostname: LAPTOP-20DEMINH
2021-02-19 16:45:24.597245: I tensorflow/core/platform/cpu_feature_guard.cc:142] This TensorFlow binary is optimized with oneAPI Deep Neural Network Library (oneDNN)to use the following C
PU instructions in performance-critical operations:  AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
2021-02-19 16:45:24.613540: I tensorflow/compiler/xla/service/service.cc:168] XLA service 0x1caab47a680 initialized for platform Host (this does not guarantee that XLA will be used). Devi
ces:
2021-02-19 16:45:24.613731: I tensorflow/compiler/xla/service/service.cc:176]   StreamExecutor device (0): Host, Default Version
WARNING:tensorflow:Error in loading the saved optimizer state. As a result, your model is starting with a freshly initialized optimizer.
[ WARN:0] global C:\Users\appveyor\AppData\Local\Temp\1\pip-req-build-k8sx3e60\opencv\modules\videoio\src\cap_msmf.cpp (435) `anonymous-namespace'::SourceReaderCB::~SourceReaderCB termina
ting async callback
```

Figure 4.5: Display Voice Output

Figure 4.5 describes the converting of text to sound and display voice output.

5.CONCLUSION AND FUTURE SCOPE

5.1 CONCLUSION

The project is a simple demonstration of how CNN can be used to solve computer vision problems with an extremely high degree of accuracy. A finger spelling sign language translator is obtained which has an accuracy of 99.98%. The project can be extended to other sign languages by building the corresponding dataset and training the CNN. The main objective has been achieved, that is, the need for an interpreter has been eliminated. There are a few finer points that need to be considered when we are running the project. The thresh needs to be monitored so that we don’t get distorted gray scales in the frames. If this issue is encountered, we need to either reset the histogram or look for places with suitable lighting conditions.The thresh needs to be monitored so that we don’t get distorted gray scales in the frames.

5.2 FUTURE SCOPE

This project can be enhanced in a few ways in the future, it could be built as a web or a mobile application for the users to conveniently access the project, also, the existing project only works for ASL, it can be extended to work for other native sign languages with enough dataset and training. This project implements a fingerspelling translator[15],[16], however, sign languages are also spoken on a contextual basis where each gesture could represent an object, verb, so, identifying this kind of contextual signing would require a higher degree of processing and natural language processing (NLP). The recent implementation of one-shot adaptation has also had success in solving real-world computer vision tasks and effectively trained deep CNN using very little domain-specific data, even as limited as single-image datasets.

BIBLIOGRAPHY

[1] Uday Patil, Saraswati Nagtilak, Sunil Rai, Swaroop Patwari, Prajay Agarwal and Rahul Sharma; ”Sign Language Translator using Deep Learning”; EasyChair Preprint; February 4, 2020.

[2] Ashish Sah and A. Arul Prakash; “Sign Language and Gesture Recognition”; EasyChair Preprint; June 2, 2020.

[3] Anna Deza and Danial Hasan; “Sign Language Recognition”; IEEE; Decemeber 2nd 2019.

[4] Vivek Bheda and N.Dianna Radpour; “Using Deep Convolutional Networks for Gesture Recognition in American Sign Language”; Department of Computer Science, Department of Linguistics State University of N ew York at Buffalo; January 25 2020.

[5] Ankit Ojha, Ayush Pandey, Shubham Maurya,Abhishek Thakur and Dr. Dayananda P, “Sign Language to Text and Speech Translation in Real Time Using Convolutional Neural Network”; International Journal of Engineering Research & Technology (IJERT); June 27 2020.

[6] Mannava Vivek, and Vitapu Gnanasagar; “Portable Sign Language Translator for Emergency Response Teams”;International Journal of Scientific Research & Engineering Trends Volume 6, Issue 3; May-June-2020.

[7] Kaushubh Jadhav, Abhishek Jaiswal , Abbas Munshi and Mayuresh Yerendhar; “Sign Language Recognition Using Neural Network”; IEEE; August 10 2020.

[8] Shubhendu Apoorv, Sudharshan Kumar Bhowmick and R Sakthi Prabha; “Indian sign language interpreter using image processing and machine learning”; Materials Science and Engineering 872 (2020) 012026 IOP Publishing doi:10.1088/1757-899X/872/1/012026; March 10 2020.

[9] Karen Simonyan & Andrew Zisserman; “ Very Deep Convolutional Networks for Large-Scale Image Recogntion”; ICLR, pp 730-734; 2015.

[10] LeCun, Y., Bengio, Y., & Hinton, G; “Deep learning”; vol 521 Nature, 521 pp 436-444; May 2015.

[11] Shubhendu Apoorv, Sudharshan Kumar Bhowmick and R Sakthi Prabha; "Indian sign language interpreter using image processing and machine learning"; Materials Science and Engineering 872 (2020) 012026 IOP Publishing doi:10.1088/1757-899X/872/1/012026; March 10 2020.

[12] Della Goswell, Jemina Napier and Rachel Locker McKee; "Sign Language Interpreting: Theory and Practice in Australia and New Zealand"; The Federation Press; 2006.

[13] Alper Yilmaz, Omar Javed, Mubarak Shah ; "Object Tracking : A Survey"; ACM Computing Survey, Vol 38 No.4, Article 13; December 2016.

[14] Deshmukh K.S., Shinde G. N; "An Adaptive Color Image Segmentation"; Electronic letters on Computer Vision and Image Analysis 5(4) : pp 12-23; December 2015.

[15] Soeb Hussain, Rupal Saxena, Xie Han, Jameel Ahmed Khan, Prof. Hyunchal Shin; "Hand Gesture Recognition Using Deep Learning"; IEEE International Conference for Design, pp 48-49; 2017.

[16] Siddharth S. Rautaray, Anupam Agrawal; "Real time hand gesture recognition system for dynamic "applications"; International Journal of UbiComp (IJU), Indian Institute of Information Technology Allahabad, India, Vol.3, No.1; January 2012.

[17] Rung-Huei Liang, Ming Ouhyoung; "A Real-time Continuous Alphabetic Sign Language to Speech Conversion VR System," Communications & Multimedia Lab., Computer Science and Information Engineering Dept., National Taiwan University, Taipei, Taiwan; 2010.

[18] Sahib Singh1,Dr. Vijay Kumar Banga; "Gesture control algorithm for personal computers"; ISSN: 2319 – 1163, Department of Electronics and Communication Engineering,Punjab, India; 2008.

[19] L.Kau, W. Su, P. Yu and S. Wei; "A real-time portable sign language translation system"; IEEE 58th International Midwest Symposium on Circuits and Systems (MWSCAS); Fort Collins, CO; pp. 1-4, doi: 10.1109/MWSCAS.2015.7282137; 2015.

[20] M. S. Nair, A. P. Nimitha and S. M. Idicula; "Conversion of Malayalam text to Indian sign language using synthetic animation"; ICNGIS, Kottayam, 2016, pp. 1-4, doi: 10.1109/ICNGIS.2016.7854002; 2016.

APPENDIX

SOURCE CODE

## Training of CNN model using Keras and saved by using .h5 extension.

```
import numpy as np
import pickle
import cv2, os
from glob import glob
from keras import optimizers
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Dropout
from keras.layers import Flatten
from keras.layers.convolutional import Conv2D
from keras.layers.convolutional import MaxPooling2D
from keras.utils import np_utils
from keras.callbacks import ModelCheckpoint
from keras import backend as K
K.set_image_data_format('channels_last')
os.environ['TF_CPP_MIN_LOG_LEVEL'] = '3'
def get_image_size():
        img = cv2.imread('gestures/1/100.jpg', 0)
        return img.shape
def get_num_of_classes():
        return len(glob('gestures/*'))
image_x, image_y = get_image_size()
def cnn_model():
        num_of_classes = get_num_of_classes()
        model = Sequential()
        model.add(Conv2D(16, (2,2), input_shape=(image_x, image_y, 1), activation='relu'))
        model.add(MaxPooling2D(pool_size=(2, 2), strides=(2, 2), padding='same'))
        model.add(Conv2D(32, (3,3), activation='relu'))
        model.add(MaxPooling2D(pool_size=(3, 3), strides=(3, 3), padding='same'))
        model.add(Conv2D(64, (5,5), activation='relu'))
        model.add(MaxPooling2D(pool_size=(5, 5), strides=(5, 5), padding='same'))
        model.add(Flatten())
        model.add(Dense(128, activation='relu'))
```

```
    model.add(Dropout(0.2))
    model.add(Dense(num_of_classes, activation='softmax'))
    sgd = optimizers.SGD(lr=1e-2)
    model.compile(loss='categorical_crossentropy', optimizer=sgd, metrics=['accuracy'])
    filepath="cnn_model_keras3.h5"
    checkpoint1 = ModelCheckpoint(filepath, monitor='val_acc', verbose=1, save_best_only=True, 
mode='max')
    callbacks_list = [checkpoint1]
    return model, callbacks_list
def train():
    with open("train_images", "rb") as f:
        train_images = np.array(pickle.load(f))
    with open("train_labels", "rb") as f:
        train_labels = np.array(pickle.load(f), dtype=np.int32)
    with open("val_images", "rb") as f:
        val_images = np.array(pickle.load(f))
    with open("val_labels", "rb") as f:
        val_labels = np.array(pickle.load(f), dtype=np.int32)
    train_images = np.reshape(train_images, (train_images.shape[0], image_x, image_y, 1))
    val_images = np.reshape(val_images, (val_images.shape[0], image_x, image_y, 1))
    train_labels = np_utils.to_categorical(train_labels)
    val_labels = np_utils.to_categorical(val_labels)
    print(val_labels.shape)
    model, callbacks_list = cnn_model()
    model.summary()
    model.fit(train_images, train_labels, validation_data=(val_images, val_labels), epochs=20, 
batch_size=500, callbacks=callbacks_list)
    scores = model.evaluate(val_images, val_labels, verbose=0)
    print("CNN Error: %.2f%%" % (100-scores[1]*100))
    model.save('cnn_model_keras3.h5')
train()
K.  clear_session();
```

## Setting of hand histogram by using the following program.

```
import cv2
import numpy as np
import pickle
def build_squares(img):
    x, y, w, h = 420, 140, 10, 10
    d = 10
    imgCrop = None
    crop = None
    for i in range(10):
        for j in range(5):
            if np.any(imgCrop == None):
                imgCrop = img[y:y+h, x:x+w]
            else:
                imgCrop = np.hstack((imgCrop, img[y:y+h, x:x+w]))
            #print(imgCrop.shape)
            cv2.rectangle(img, (x,y), (x+w, y+h), (0,255,0), 1)
            x+=w+d
        if np.any(crop == None):
            crop = imgCrop
        else:
            crop = np.vstack((crop, imgCrop))
        imgCrop = None
        x = 420
        y+=h+d
    return crop
def get_hand_hist():
    cam = cv2.VideoCapture(1)
    if cam.read()[0]==False:
        cam = cv2.VideoCapture(0)
    x, y, w, h = 300, 100, 300, 300
    flagPressedC, flagPressedS = False, False
    imgCrop = None
    while True:
        img = cam.read()[1]
        img = cv2.flip(img, 1)
        img = cv2.resize(img, (640, 480))
        hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)
        keypress = cv2.waitKey(1)
        if keypress == ord('c'):
            hsvCrop = cv2.cvtColor(imgCrop, cv2.COLOR_BGR2HSV)
            flagPressedC = True
            hist = cv2.calcHist([hsvCrop], [0, 1], None, [180, 256], [0, 180, 0, 256])
            cv2.normalize(hist, hist, 0, 255, cv2.NORM_MINMAX)
        elif keypress == ord('s'):
            flagPressedS = True
            break
        if flagPressedC:
            dst = cv2.calcBackProject([hsv], [0, 1], hist, [0, 180, 0, 256], 1)
            dst1 = dst.copy()
            disc = cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(10,10))
            cv2.filter2D(dst,-1,disc,dst)
            blur = cv2.GaussianBlur(dst, (11,11), 0)
            blur = cv2.medianBlur(blur, 15)
            ret,thresh = cv2.threshold(blur,0,255,cv2.THRESH_BINARY+cv2.THRESH_OTSU)
            thresh = cv2.merge((thresh,thresh,thresh))
            #cv2.imshow("res", res)
            cv2.imshow("Thresh", thresh)
        if not flagPressedS:
            imgCrop = build_squares(img)
        #cv2.rectangle(img, (x,y), (x+w, y+h), (0,255,0), 2)
        cv2.imshow("Set hand histogram", img)
    cam.release()
    cv2.destroyAllWindows()
    with open("hist", "wb") as f:
        pickle.dump(hist, f)
get_hand_hist()
```

## Gesture recognition and translation by using the following program.

```
import cv2, pickle
import numpy as np
import tensorflow as tf
from cnn_tf import cnn_model_fn
import os
import sqlite3, pyttsx3
from keras.models import load_model
from threading import Thread
engine = pyttsx3.init()
engine.setProperty('rate', 150)
os.environ['TF_CPP_MIN_LOG_LEVEL'] = '3'
model = load_model('cnn_model_keras2.h5')
def get_hand_hist():
        with open("hist", "rb") as f:
                hist = pickle.load(f)
        return hist
def get_image_size():
        img = cv2.imread('gestures/0/100.jpg', 0)
        return img.shape
image_x, image_y = get_image_size()
def keras_process_image(img):
        img = cv2.resize(img, (image_x, image_y))
        img = np.array(img, dtype=np.float32)
        img = np.reshape(img, (1, image_x, image_y, 1))
        return img
def keras_predict(model, image):
        processed = keras_process_image(image)
        pred_probab = model.predict(processed)[0]
        pred_class = list(pred_probab).index(max(pred_probab))
        return max(pred_probab), pred_class
def get_pred_text_from_db(pred_class):
        conn = sqlite3.connect("gesture_db.db")
        cmd = "SELECT g_name FROM gesture WHERE g_id="+str(pred_class)
        cursor = conn.execute(cmd)
        for row in cursor:
                return row[0]
def get_pred_from_contour(contour, thresh):
        x1, y1, w1, h1 = cv2.boundingRect(contour)
        save_img = thresh[y1:y1+h1, x1:x1+w1]
        text = ""
        if w1 > h1:
                save_img = cv2.copyMakeBorder(save_img, int((w1-h1)/2) , int((w1-h1)/2) , 0, 0,
cv2.BORDER_CONSTANT, (0, 0, 0))
        elif h1 > w1:
                save_img = cv2.copyMakeBorder(save_img, 0, 0, int((h1-w1)/2) , int((h1-w1)/2) ,
cv2.BORDER_CONSTANT, (0, 0, 0))
        pred_probab, pred_class = keras_predict(model, save_img)
        if pred_probab*100 > 70:
                text = get_pred_text_from_db(pred_class)
        return text
def get_operator(pred_text):
        try:
                pred_text = int(pred_text)
        except:
                return ""
        operator = ""
        if pred_text == 1:
                operator = "+"
        elif pred_text == 2:
                operator = "-"
        elif pred_text == 3:
                operator = "*"
        elif pred_text == 4:
                operator = "/"
        elif pred_text == 5:
                operator = "%"
        elif pred_text == 6:
                operator = "**"
        elif pred_text == 7:
                operator = ">>"
        elif pred_text == 8:
                operator = "<<"
        elif pred_text == 9:
                operator = "&"
        elif pred_text == 0:
                operator = "|"
        return operator
hist = get_hand_hist()
x, y, w, h = 300, 100, 300, 300
```

```
is_voice_on = True
def get_img_contour_thresh(img):
        img = cv2.flip(img, 1)
        imgHSV = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)
        dst = cv2.calcBackProject([imgHSV], [0, 1], hist, [0, 180, 0, 256], 1)
        disc = cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(10,10))
        cv2.filter2D(dst,-1,disc,dst)
        blur = cv2.GaussianBlur(dst, (11,11), 0)
        blur = cv2.medianBlur(blur, 15)
        thresh = cv2.threshold(blur,0,255,cv2.THRESH_BINARY+cv2.THRESH_OTSU)[1]
        thresh = cv2.merge((thresh,thresh,thresh))
        thresh = cv2.cvtColor(thresh, cv2.COLOR_BGR2GRAY)
        thresh = thresh[y:y+h, x:x+w]
        contours = cv2.findContours(thresh.copy(), cv2.RETR_TREE, cv2.CHAIN_APPROX_NONE)[0]
        return img, contours, thresh
def say_text(text):
        if not is_voice_on:
                return
        while engine._inLoop:
                pass
        engine.say(text)
        engine.runAndWait()
def calculator_mode(cam):
        global is_voice_on
        flag = {"first": False, "operator": False, "second": False, "clear": False}
        count_same_frames = 0
        first, operator, second = "", "", ""
        pred_text = ""
        calc_text = ""
        info = "Enter first number"
        Thread(target=say_text, args=(info,)).start()
        count_clear_frames = 0
        while True:
                img = cam.read()[1]
                img = cv2.resize(img, (640, 480))
                img, contours, thresh = get_img_contour_thresh(img)
                old_pred_text = pred_text
                if len(contours) > 0:
                        contour = max(contours, key = cv2.contourArea)
                        if cv2.contourArea(contour) > 10000:
```

```
pred_text = get_pred_from_contour(contour, thresh)
if old_pred_text == pred_text:
    count_same_frames += 1
else:
    count_same_frames = 0
if pred_text == "C":
    if count_same_frames > 5:
        count_same_frames = 0
        first, second, operator, pred_text, calc_text = '', '', '', '', ''
        flag['first'], flag['operator'], flag['second'], flag['clear'] =
False, False, False, False
        info = "Enter first number"
        Thread(target=say_text, args=(info,)).start()
elif pred_text == "Best of Luck " and count_same_frames > 15:
    count_same_frames = 0
    if flag['clear']:
        first, second, operator, pred_text, calc_text = '', '', '', '', ''
        flag['first'], flag['operator'], flag['second'], flag['clear'] =
False, False, False, False
        info = "Enter first number"
        Thread(target=say_text, args=(info,)).start()
    elif second != '':
        flag['second'] = True
        info = "Clear screen"
        #Thread(target=say_text, args=(info,)).start()
        second = ''
        flag['clear'] = True
        try:
            calc_text += "= "+str(eval(calc_text))
        except:
            calc_text = "Invalid operation"
        if is_voice_on:
            speech = calc_text
            speech = speech.replace('-', ' minus ')
            speech = speech.replace('/', ' divided by ')
            speech =speech.replace('**',' raised to the power ')
            speech = speech.replace('*', ' multiplied by ')
            speech = speech.replace('%', ' mod ')
            speech = speech.replace('>>', ' bitwise right shift ')
            speech = speech.replace('<<', ' bitwise leftt shift ')
```

```
                                                speech = speech.replace('&', ' bitwise and ')
                                                speech = speech.replace('|', ' bitwise or ')
                                                Thread(target=say_text, args=(speech,)).start()
                                elif first != '':
                                        flag['first'] = True
                                        info = "Enter operator"
                                        Thread(target=say_text, args=(info,)).start()
                                        first = ''
                        elif pred_text != "Best of Luck " and pred_text.isnumeric():
                                if flag['first'] == False:
                                        if count_same_frames > 15:
                                                count_same_frames = 0
                                                Thread(target=say_text, args=(pred_text,)).start()
                                                first += pred_text
                                                calc_text += pred_text
                                elif flag['operator'] == False:
                                        operator = get_operator(pred_text)
                                        if count_same_frames > 15:
                                                count_same_frames = 0
                                                flag['operator'] = True
                                                calc_text += operator
                                                info = "Enter second number"
                                                Thread(target=say_text, args=(info,)).start()
                                                operator = ''
                                elif flag['second'] == False:
                                        if count_same_frames > 15:
                                                Thread(target=say_text, args=(pred_text,)).start()
                                                second += pred_text
                                                calc_text += pred_text
                                                count_same_frames = 0

                if count_clear_frames == 30:
                        first, second, operator, pred_text, calc_text = '', '', '', '', ''
                        flag['first'], flag['operator'], flag['second'], flag['clear'] = False, False, False, False
                        info = "Enter first number"
                        Thread(target=say_text, args=(info,)).start()
                        count_clear_frames = 0
                blackboard = np.zeros((480, 640, 3), dtype=np.uint8)
                cv2.putText(blackboard, "Calculator Mode", (100, 50), cv2.FONT_HERSHEY_TRIPLEX, 1.5,
(255, 0,0))
```

```
                cv2.putText(blackboard, "Predicted text- " + pred_text, (30, 100), 
cv2.FONT_HERSHEY_TRIPLEX, 1, (255, 255, 0))
                cv2.putText(blackboard, "Operator " + operator, (30, 140), cv2.FONT_HERSHEY_TRIPLEX, 
1, (255, 255, 127))
                cv2.putText(blackboard, calc_text, (30, 240), cv2.FONT_HERSHEY_TRIPLEX, 2, (255, 255, 
255))
                cv2.putText(blackboard, info, (30, 440), cv2.FONT_HERSHEY_TRIPLEX, 1, (0, 255, 255) )
                if is_voice_on:
                        cv2.putText(blackboard, "Voice on", (450, 440), cv2.FONT_HERSHEY_TRIPLEX, 
1, (255, 127, 0))
                else:
                        cv2.putText(blackboard, "Voice off", (450, 440), cv2.FONT_HERSHEY_TRIPLEX, 
1, (255, 127, 0))
                cv2.rectangle(img, (x,y), (x+w, y+h), (0,255,0), 2)
                res = np.hstack((img, blackboard))
                cv2.imshow("Recognizing gesture", res)
                cv2.imshow("thresh", thresh)
                keypress = cv2.waitKey(1)
                if keypress == ord('q') or keypress == ord('t'):
                        break
                if keypress == ord('v') and is_voice_on:
                        is_voice_on = False
                elif keypress == ord('v') and not is_voice_on:
                        is_voice_on = True
        if keypress == ord('t'):
                return 1
        else:
                return 0
def text_mode(cam):
        global is_voice_on
        text = ""
        word = ""
        count_same_frame = 0
        while True:
                img = cam.read()[1]
                img = cv2.resize(img, (640, 480))
                img, contours, thresh = get_img_contour_thresh(img)
                old_text = text
                if len(contours) > 0:
                        contour = max(contours, key = cv2.contourArea)
```

```
            if cv2.contourArea(contour) > 10000:
                text = get_pred_from_contour(contour, thresh)
                if old_text == text:
                    count_same_frame += 1
                else:
                    count_same_frame = 0
                if count_same_frame > 20:
                    if len(text) == 1:
                        Thread(target=say_text, args=(text, )).start()
                    word = word + text
                    if word.startswith('I/Me '):
                        word = word.replace('I/Me ', 'I ')
                    elif word.endswith('I/Me '):
                        word = word.replace('I/Me ', 'me ')
                    count_same_frame = 0
            elif cv2.contourArea(contour) < 1000:
                if word != '':
                    Thread(target=say_text, args=(word, )).start()
                text = ""
                word = ""
        else:
            if word != '':
                #print('yolo1')
                #say_text(text)
                Thread(target=say_text, args=(word, )).start()
            text = ""
            word = ""
        blackboard = np.zeros((480, 640, 3), dtype=np.uint8)
        cv2.putText(blackboard, "Text Mode", (180, 50), cv2.FONT_HERSHEY_TRIPLEX, 1.5, (255,
0,0))
        cv2.putText(blackboard, "Predicted text- " + text, (30, 100), cv2.FONT_HERSHEY_TRIPLEX,
1, (255, 255, 0))
        cv2.putText(blackboard, word, (30, 240), cv2.FONT_HERSHEY_TRIPLEX, 2, (255, 255,
255))
        if is_voice_on:
            cv2.putText(blackboard, "Voice on", (450, 440), cv2.FONT_HERSHEY_TRIPLEX,
1, (255, 127, 0))
        else:
            cv2.putText(blackboard, "Voice off", (450, 440), cv2.FONT_HERSHEY_TRIPLEX,
1, (255, 127, 0))
```

```
                    cv2.rectangle(img, (x,y), (x+w, y+h), (0,255,0), 2)
                    res = np.hstack((img, blackboard))
                    cv2.imshow("Recognizing gesture", res)
                    cv2.imshow("thresh", thresh)
                    keypress = cv2.waitKey(1)
                    if keypress == ord('q') or keypress == ord('c'):
                              break
                    if keypress == ord('v') and is_voice_on:
                              is_voice_on = False
                    elif keypress == ord('v') and not is_voice_on:
                              is_voice_on = True
          if keypress == ord('c'):
                    return 2
          else:
                    return 0
def recognize():
          cam = cv2.VideoCapture(1)
          if cam.read()[0]==False:
                    cam = cv2.VideoCapture(0)
          text = ""
          word = ""
          count_same_frame = 0
          keypress = 1
          while True:
                    if keypress == 1:
                              keypress = text_mode(cam)
                    elif keypress == 2:
                              keypress = calculator_mode(cam)
                    else:
                              break
keras_predict(model, np.zeros((50, 50), dtype = np.uint8))
recognize()
```