Sign Language Illustrator

Submitted in partial fulfillment of the requirement for the award of Degree of Bachelor of Technology in Information Technology Discipline

Submitted To

SVKM's NMIMS,
Mukesh Patel School of Technology Management & Engineering,
Shirpur Campus (M.H.)

Submitted By:

Prateek Dembla – 70011117020
Sunidhi Manish Jain – 70011117031
Rubin Jhambani – 70011117035
Ojesh Vyas – 70011117073

Under the Supervision Of:

Mr. Prashant Udawant
(Asst. Professor, Information Technology)

DEPARTMENT OF INFORMATION TECHNOLOGY
Mukesh Patel School of Technology Management & Engineering
SESSION: 2020-21

CERTIFICATE

This is to certify that the work embodies in this Project entitled **"Sign Language Illustrator"** being submitted by

Prateek Dembla - 70011117020
Sunidhi Manish Jain - 70011117031
Rubin Jhambani - 70011117035
Ojesh Vyas - 70011117073

for partial fulfillment of the requirement for the award of "**Bachelor of Technology** in **Information Technology"** discipline to "SVKM's NMIMS, Mumbai (M.H.)" during the academic year 2020-21 is a record of bonafide piece of work, carried out by him under my supervision and guidance in the **"Department of Information Technology", MPSTME, Shirpur (M.H.).**

APPROVED & SUPERVISED BY:

Mr. Prashant Udawant
(Asst. Professor, Information Technology)

FORWARDED BY:

(Dr. Nitin Choubey)
H.O.D., IT
MPSTME, Shirpur Campus

(Dr. Kamal Mehta)
Associate Dean,
MPSTME, Shirpur Campus

DEPARTMENT OF INFORMATION TECHNOLOGY
Mukesh Patel School of Technology Management & Engineering

CERTIFICATE OF APPROVAL

The Project entitled **“Sign Language Illustrator”** being submitted by

Prateek Dembla - 70011117020

Sunidhi Manish Jain - 70011117031

Rubin Jhambani - 70011117035

Ojesh Vyas - 70011117073

has been examined by us and is hereby approved for the award of degree **“Bachelor of Technology in Information Technology Discipline”**, for which it has been submitted. It is understood that by this approval the undersigned do not necessarily endorse or approve any statement made, opinion expressed or conclusion drawn therein, but approve the project only for the purpose for which it has been submitted.

(Internal Examiner) **(External Examiner)**

Date: **Date:**

DEPARTMENT OF INFORMATION TECHNOLOGY

Mukesh Patel School of Technology Management & Engineering

<u>DECLARATION</u>

We,

Prateek Dembla

Sunidhi Manish Jain

Rubin Jhambani

Ojesh Vyas

The students of **Bachelor of Technology** in **Information Technology discipline**, **Session: 2020-21, MPSTME, Shirpur Campus,** hereby declare that the work presented in this Project entitled **“Sign Language Illustrator”** is the outcome of our work, is bonafide and correct to the best of our knowledge and this work has been carried out taking care of Engineering Ethics. The work presented does not infringe any patented work and has not been submitted to any other university or anywhere else for the award of any degree or any professional diploma.

Prateek dembla -70011117020
Sunidhi Manish Jain - 70011117031
Rubin Jhambani - 70011117035
Ojesh Vyas - 70011117073

Date:

ACKNOWLEDGEMENT

After the completion of this Major Project work, words are not enough to express my feelings about all those who helped me to reach my goal; feeling above this is my indebtedness to The Almighty for providing me this moment in life.

It’s a great pleasure and moment of immense satisfaction for me to express my profound gratitude to **Prof. Prashant Udawant**, **Asst. Professor**, Information Technology Department, MPSTME, Shirpur, whose constant encouragement enabled me to work enthusiastically. Their perpetual motivation, patience and excellent expertise in discussion during progress of the project work have benefited me to an extent, which is beyond expression. Their depth and breadth of knowledge of the Information Technology field made me realize that theoretical knowledge always helps to develop efficient operational software, which is a blend of all core subjects of the field. I am highly indebted to them for their invaluable guidance and ever-ready support in the successful completion of this project in time. Working under their guidance has been a fruitful and unforgettable experience.

We express my sincere thanks and gratitude to Dr. Nitin Choubey, Head of Department, Information Technology Department, MPSTME, Shirpur, for providing necessaryinfrastructure and help to complete the project work successfully.

We also extend my deepest gratitude to Dr. Akshay Malhotra, Director, SVKM’S NMIMS, Shirpur Campus and Dr. Kamal Mehta, Associate Dean, SVKM’S NMIMS, Shirpur Campus for providing all the necessary facilities and true encouraging environment to bring out the best of my endeavors.

We sincerely wish to express my grateful thanks to all members of the staff of the Information Technology department and all those who have embedded me with technical knowledge of computer technology during various stages of B.Tech. Information Technology.

We would like to acknowledge all my friends, who have contributed directly or indirectly in this Major Project work.

The successful completion of a Major Project is generally not an individual effort. It is an outcome of the cumulative effort of a number of persons, each having their own importance to the objective. This section is a vote of thanks and gratitude towards all those persons who have directly or indirectly contributed in their own special way towards the completion of this project.

Prateek Dembla -70011117020
Sunidhi Manish Jain - 70011117031
Rubin Jhambani - 70011117035
Ojesh Vyas - 70011117073

ABSTRACT

Sign language is one of the oldest and most natural forms of language for communication, but since most people do not know sign language and interpreters are very difficult to come by we have come up with a real time method using neural networks for fingerspelling based American Sign Language. The term sign language illustrator represents the device throughout. It is a suggested structure aimed at reducing the disparity in language between ordinary and deaf and dumb people. It is entirely focused on the theory of transforming images and machine learning. The gesture image is filmed in this process and then pre-processed and compared to the data collection that finally gives our output, i.e., in text format, the significance of the gesture. and a hand motion recognition method developed using the principle of machine learning and neural networking, since this approach is more practical and can achieve optimum precision. Other solutions that use HD cameras or sensor-based sensors that detect hand movements are expensive and need more hardware in addition.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

American sign language is a predominant sign language. Since the only disability D&M people have is communication related and they cannot use spoken languages hence the process of exchange of thoughts and messages in various ways such as speech, signals,behavior and visuals. Deaf and dumb (D&M) people make use of their hands to express different gestures to express their ideas with other people. Gestures are the nonverbally exchanged messages and these gestures are understood with vision. This nonverbal communication of deaf and dumb people is called sign language.

In our project we basically focus on producing a model which can recognize Fingerspelling based hand gestures in order to form a complete word by combining each gesture. The gestures we aim to train are as given in the image below.

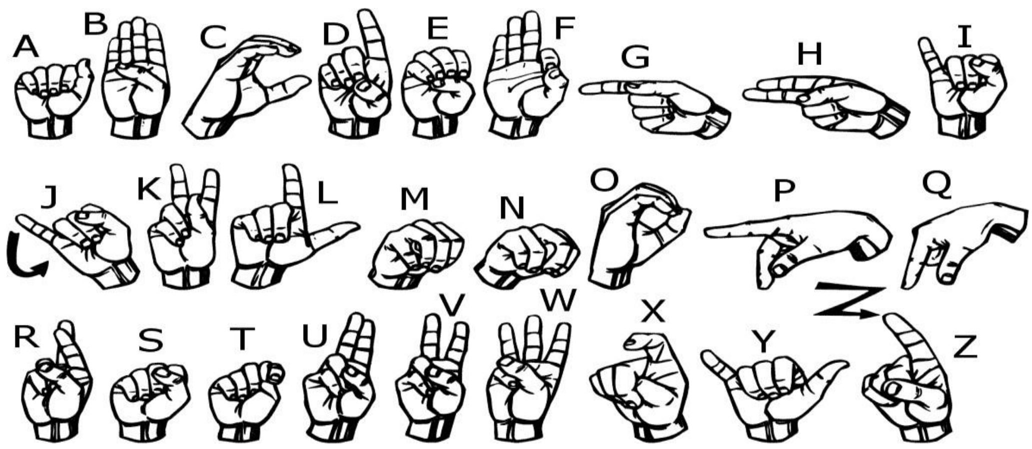

FIG 1.1 Gestures of respective alphabets

1.1 Purpose

The aim is to develop a user-friendly human computer interface (HCI) where the computer understands the human sign language. There are various sign languages all over the world, namely American Sign Language (ASL), French Sign Language, British Sign Language (BSL), Indian Sign language, Japanese Sign Language and work has been done on other languages all around the world.

1.2 Scope

For interaction between normal people and D&M people a language barrier is created as sign language structure which is different from normal text. So, they depend on vision-based communication for interaction. If there is a common interface that converts the sign language to text the gestures can be easily understood by the other people. So, research has been made for a vision-based interface system where D&M people can enjoy communication without really knowing each other's language.

CHAPTER 2

LITERATURE SURVEY

In recent years there has been tremendous research done on hand gesture recognition. With the help of literature survey done we realized the basic steps in hand gesture recognition are: -

● Data acquisition
● Data preprocessing
● Feature extraction
● Gesture classification

2.1 Data acquisition:

The different approaches to acquire data about the hand gesture can be done in the following ways:

2.1.1. Use of sensory devices

It uses electromechanical devices to provide exact hand configuration, and position. Different glove-based approaches can be used to extract information. But it is expensive and not user friendly.

2.1.2. Vision based approach

In vision-based methods, a computer camera is the input device for observing the information of hands or fingers. The Vision Based methods require only a camera, thus realizing a natural interaction between humans and computers without the use of any extra devices. These systems tend to complement biological vision by describing artificial vision systems that are implemented in software and/or hardware. The main challenge of vision-based hand detection is to cope with the large variability of human hand's appearance due to a huge number of hand movements, to different skin-color possibilities as well as to the variations in viewpoints, scales, and speed of the camera capturing the scene.

2.2. Data preprocessing and Feature extraction

● In [1] the approach for hand detection combines threshold-based color detection with background subtraction. We can use Adaboost face detector to differentiate between faces and hands as both involve similar skin-color.

● We can also extract necessary image which is to be trained by applying a filter called Gaussian blur. The filter can be easily applied using open computer vision also known as OpenCV and is described in [3].

● For extracting necessary image which is to be trained we can use instrumented gloves as mentioned in [4]. This helps reduce computation time for preprocessing and can give us more concise and accurate data compared to applying filters on data received from video extraction.

● We tried doing the hand segmentation of an image using color segmentation techniques but as mentioned in the research paper skin color and tone is highly dependent on the lighting conditions due to which output, we got for the segmentation we tried to do were no so great. Moreover, we have a huge number of symbols to be trained for our project many of which look similar to each other like the gesture for symbol ‘V’ and digit ‘2’, hence we decided that in order to produce better accuracies for our large number of symbols, rather than segmenting the hand out of a random background we keep background of hand a stable single color so that we don’t need to segment it on the basis of skin color. This would help us to get better results.

2.3 Gesture classification:

● In [1] Hidden Markov Models (HMM) is used for the classification of the gestures. This model deals with dynamic aspects of gestures. Gestures are extracted from a sequence of video images by tracking the skin-color blobs corresponding to the hand into a body– face space centered on the face of the user. The goal is to recognize two classes of gestures: deictic and symbolic. The image is filtered using a fast look–up indexing table. After filtering, skin color pixels are gathered into blobs. Blobs are statistical objects based on the location (x,y) and the colourimetry (Y,U,V) of the skin color pixels in order to determine homogeneous areas.

● In [2] Naïve Bayes Classifier is used which is an effective and fast method for static hand gesture recognition. It is based on classifying the different gestures according to geometric based invariants which are obtained from image data after segmentation. Thus, unlike many other recognition methods, this method is not dependent on skin color. The gestures are extracted from each frame of the video, with a static background. The first step is to segment and label the objects of interest and to extract geometric invariants from them. Next step is the classification of gestures by using a K nearest neighbor algorithm aided with distance weighting algorithm (KNNDW) to provide suitable data for a locally weighted Naïve Bayes‟ classifier.

● According to paper on "Human Hand Gesture Recognition Using a Convolution Neural Network" by Hsien-I Lin, Ming-Hsiang Hsu, and Wei-Kai Chen graduates of Institute of Automation Technology National Taipei University of Technology Taipei, Taiwan, they construct a skin model to extract the hand out of an image and then apply binary threshold to the whole image. After obtaining the threshold image they calibrate it about the principal axis in order to center the image about it. They input this image to a convolutional neural network model in order to train and predict the outputs. They have trained their model over 7 hand gestures and using their model they produce an accuracy of around 95% for those 7 gestures.

CHAPTER 3
PROBLEM DEFINITION AND PROPOSED SOLUTION

3.1 Problem Definition

Sign Language Illustrator is based on concept of Image processing. In recent year there is lot of research on gesture recognition using Kinect sensor on using HD camera but camera and Kinect sensors are more costly.

This proposed system is focused on reduce cost and improve robustness of the proposed system using simple web camera and neural networks to monitor and capture the patient’s gestures using image processing and computer vision so that it can help them to convey their thoughts and feelings.

3.2 Proposed Solution

The system is a vision-based approach. All the signs are represented with bare hands and so it eliminates the problem of using any artificial devices for interaction.

3.2.1 Data Set Generation

For the project we tried to find already made datasets but we couldn’t find dataset in the form of raw images that matched our requirements. All we could find were the datasets in the form of RGB values. Hence, we decided to create our own data set.

Steps we followed to create our data set are as follows.

We used Open computer vision (OpenCV) library in order to produce our dataset. Firstly, we captured around 900 images of each of the symbol in ASL for training purposes and around 200 images per symbol for testing purpose.

First, we capture each frame shown by the webcam of our machine. In each frame we define a region of interest (ROI) which is denoted by a blue bounded square as shown in the image below.

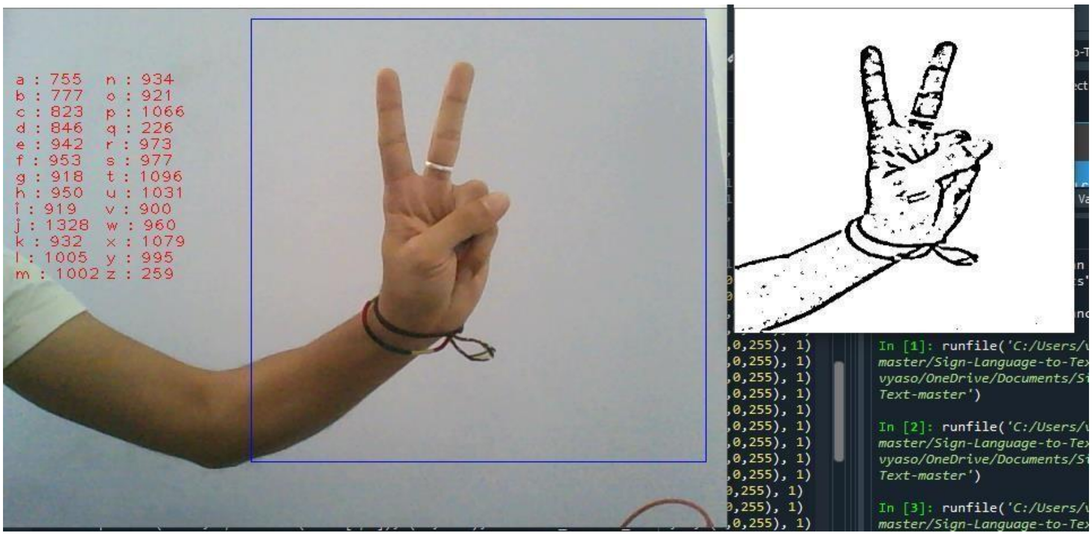

FIG 3.1 Capturing the dataset

From this whole image we extract our ROI which is RGB and convert it into gray scale Image. Finally, we apply our gaussian blur filter to our image which helps us extract various features of our image. The image after applying gaussian blur looks like below.

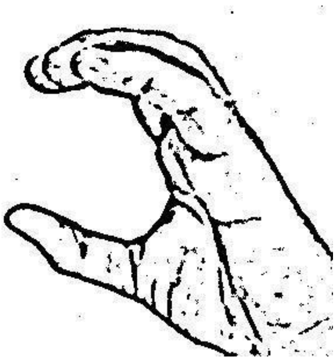

FIG 3.2 Image after applying gaussian blur

3.2.2 GESTURE CLASSIFICATION

The approach which we used for this project is:

Our approach uses two layers of algorithms to predict the final symbol of the user.

Algorithm Layer 1:

1. Apply gaussian blur filter and threshold to the frame taken with OpenCV to get the processed image after feature extraction.

2. This processed image is passed to the CNN model for prediction and if a letter is detected for more than 50 frames then the letter is printed and taken into consideration for forming the word.
3. Spaces between the words are considered using the blank symbol.

Algorithm Layer 2:

1. We detect various sets of symbols which show similar results on getting detected.
2. We then classify between those sets using classifiers made for those sets only.

Layer 1:

CNN Model:

1. 1st Convolution Layer:

The input picture has a resolution of 128x128 pixels. It is first processed in the first convolutional layer using 32 filter weights (3x3 pixels each). this will result in a 126X126 pixel image, one for each Filter-weights.

2. 1st Pooling Layer:

The pictures are down sampled using max pooling of2x2 i.e., we keep the highest value in the 2x2 square of array. Therefore, our picture is down sampled to 63x63 pixels.

3. 2nd Convolution Layer:

Now, these 63 x 63 from the output of the first pooling layer is served as an input to the second convolutional layer. It is processed in the second convolutional layer using 2 filter weights (3x3 pixels each). This will result in a 60 x 60-pixel image.

4. 2nd Pooling Layer:

The resulting images are down sampled again using a max pool of 2x2 and is reduced to 30 x 30 resolution of images.

5. 1st Densely Connected Layer:

Now these images are used as an input to a fully connected layer with 128neurons and the output from the second convolutional layer is reshaped to an array of 30x30x32 =28800 values. The input to this layer is an array of28800 values. The output of these layer is fed to the 2nd Densely Connected Layer. We are using a dropout layer of value 0.5 to avoid overfitting.

6. 2nd Densely Connected Layer:
Now the output from the 1st Densely Connected Layer is used as an input to a fully connected layer with 96 neurons.

7. Final layer:
The output of the 2nd Densely Connected Layer serves as an input for the final layer which will have the number of neurons as the number of classes we are classifying (alphabets + blank symbol).

Activation Function:

We have used ReLu (Rectified Linear Unit) in each of the layers (convolutional as well as fully connected neurons). ReLu calculates max(x,0) for each input pixel. This adds nonlinearity to the formula and helps to learn more. REL helps in removing the vanishing gradient problem and speeding up the training by reducing the computation time.

Pooling Layer:

We apply Max pooling to the input image with a pool size of (2, 2) with relu activation function. This reduces the amount of parameters thus lessening the computation cost and reduces overfitting.

Dropout Layers:

The problem of overfitting, where after training, the weights of the network are so tuned to the training examples they are given that the network doesn’t perform well when given new examples. This layer “drops out” a random set of activations in that layer by setting them to zero. The network should be able to provide the right classification or output for a specific example even if some of the activations are dropped out [5].

Optimizer:

We have used Adam optimizer for updating the model in response to the output of the loss function. Adam combines the advantages of two extensions of two stochastic gradient descent algorithms namely adaptive gradient algorithm (ADA GRAD) and root mean square propagation (RMSProp)

Layer 2:

We are using two layers of algorithms to verify and predict symbols which are more similar to each other so that we can get us close as we can get to detect the symbol shown. In our testing we found that following symbols were not showing properly and were giving other symbols also:

1. For D: R and U
2. For U: D and R
3. For I: T, D, K and I
4. For S: M and N

So, to handle above cases we made three different classifiers for classifying these sets:

1. {D, R, U}
2. {T, K, D, I}
3. {S, M, N}

3.2.3 Implementation:

Finger spelling sentence formation:

1. Whenever the count of a letter detected exceeds a specific value and no other letter is close to it by a threshold, we print the letter and add it to the current string (In our code we kept the value as 50 and difference threshold as 20).

2. Otherwise we clear the current dictionary which has the count of detections of presentsymbol to avoid the probability of a wrong letter getting predicted.

3. Whenever the count of a blank (plain background) detected exceeds a specific value and if the current buffer is empty no spaces are detected.

4. In other case it predicts the end of word by printing a space and the current gets appended to the sentence below.

Autocorrect Feature:

A python library Hunspell_suggest is used to suggest correct alternatives for each (incorrect) input word and we display a set of words matching the current word in which the user can select a word to append it to the current sentence. This helps in reducing mistakes committed in spellings and assists in predicting complex words.

CHAPTER 4

DESIGN

4.1 Architecture diagram

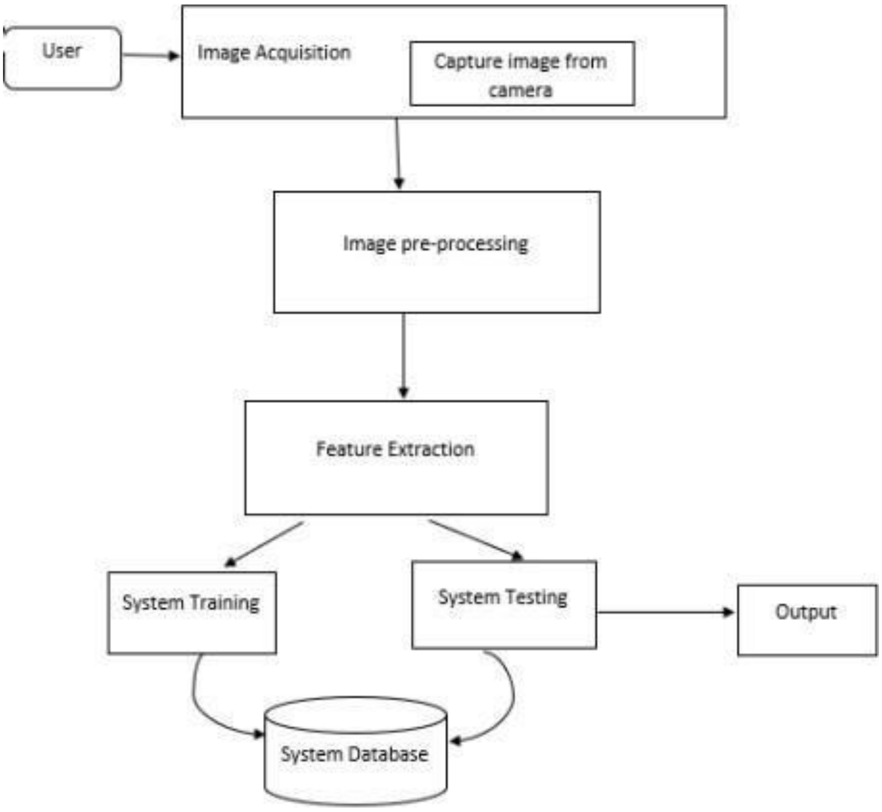

FIG 4.1: Architectural design of hand gesture recognition system

The design of gesture recognition system essentially involves the following four aspects:

1) Image acquisition
2) Image pre-processing
3) Feature extraction
4) Gesture recognition (output)

Taking this into account, a possible solution to be used in any vision-based hand recognition system for human-computer interaction is represented in the following diagram.

Components of Architectural design system:

4.1.1 Image Acquisition

Image Acquisition is accomplished by means of a camera, which captures images frame by frame

4.1.2 Image Pre-processing

The main function of this phase is to extract the hand image from its background. It involves color filter, noise filter, smoothing

4.1.3 Feature Extraction

This phase finds and extracts features of hand image. Then the system is being trained which is then stored in a database. For testing, data from the database is being taken to compare the hand gesture stored in the database.

4.1.4 Gesture recognition

If the image is matched with the database, then the gesture is recognized and we get a text message as an output for it.

4.2 Data Flow Diagram

4.2.1 DFD Level 0

In DFD Level 0, user gives input image to the system using a web-cam. Application performs required operations on the image.

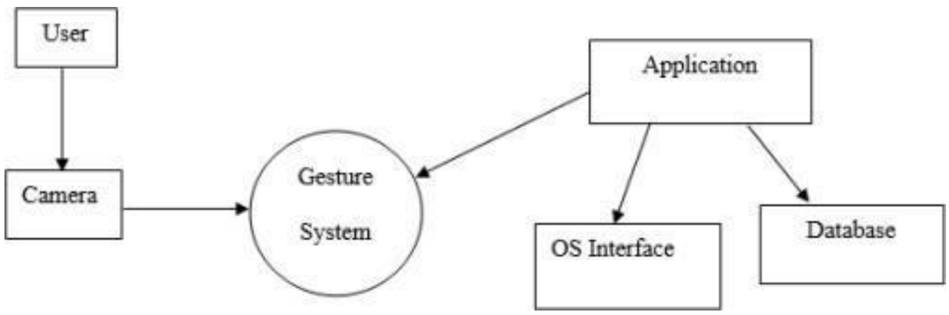

FIG 4.2.1 Data flow diagram Level 0

4.2.2 DFD Level 1

In DFD Level 1, hand gestures are captured through a webcam. Processing is done on hand gestures to detect events of hand.

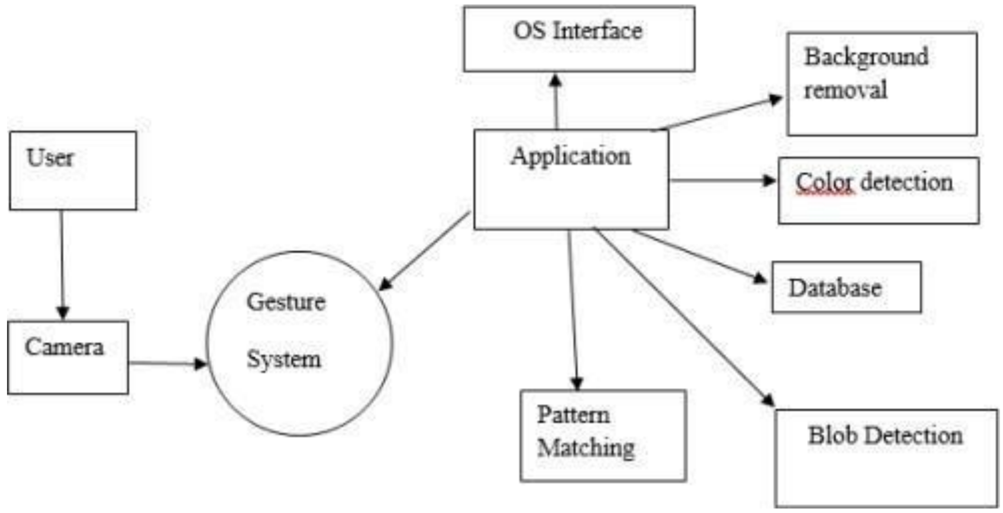

FIG 4.2.2 Data flow diagram Level 1

4.3 Use case Diagram

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved.

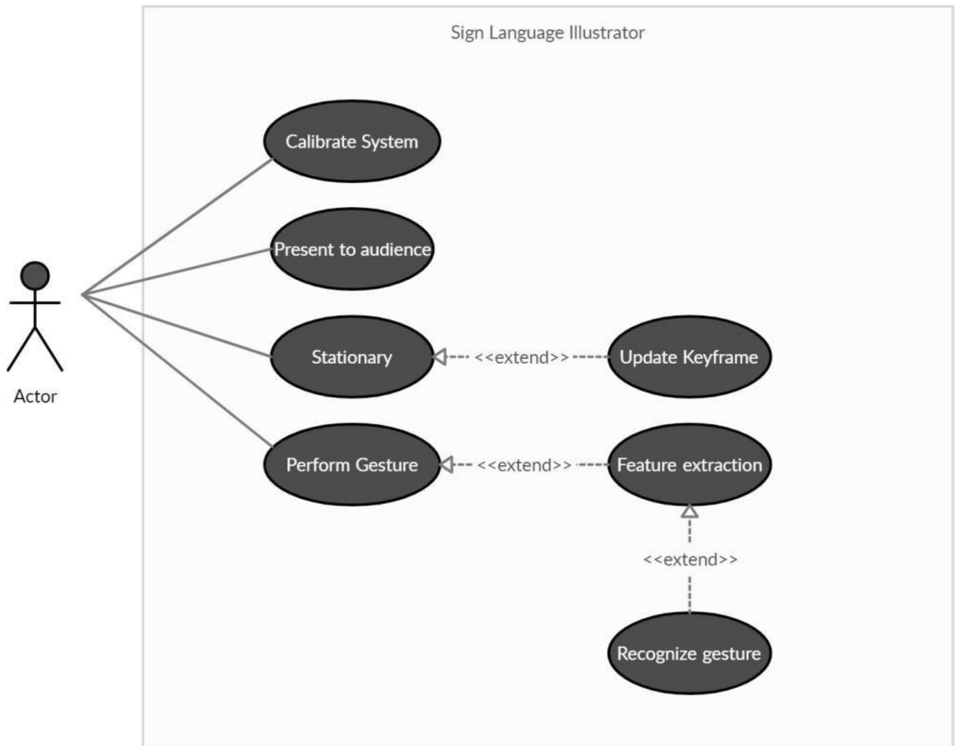

FIG 4.3 Use Case design of hand gesture recognition system

4.4 Activity Diagram

Activity Diagram is basically a flowchart to represent the flow from one activity to another activity.

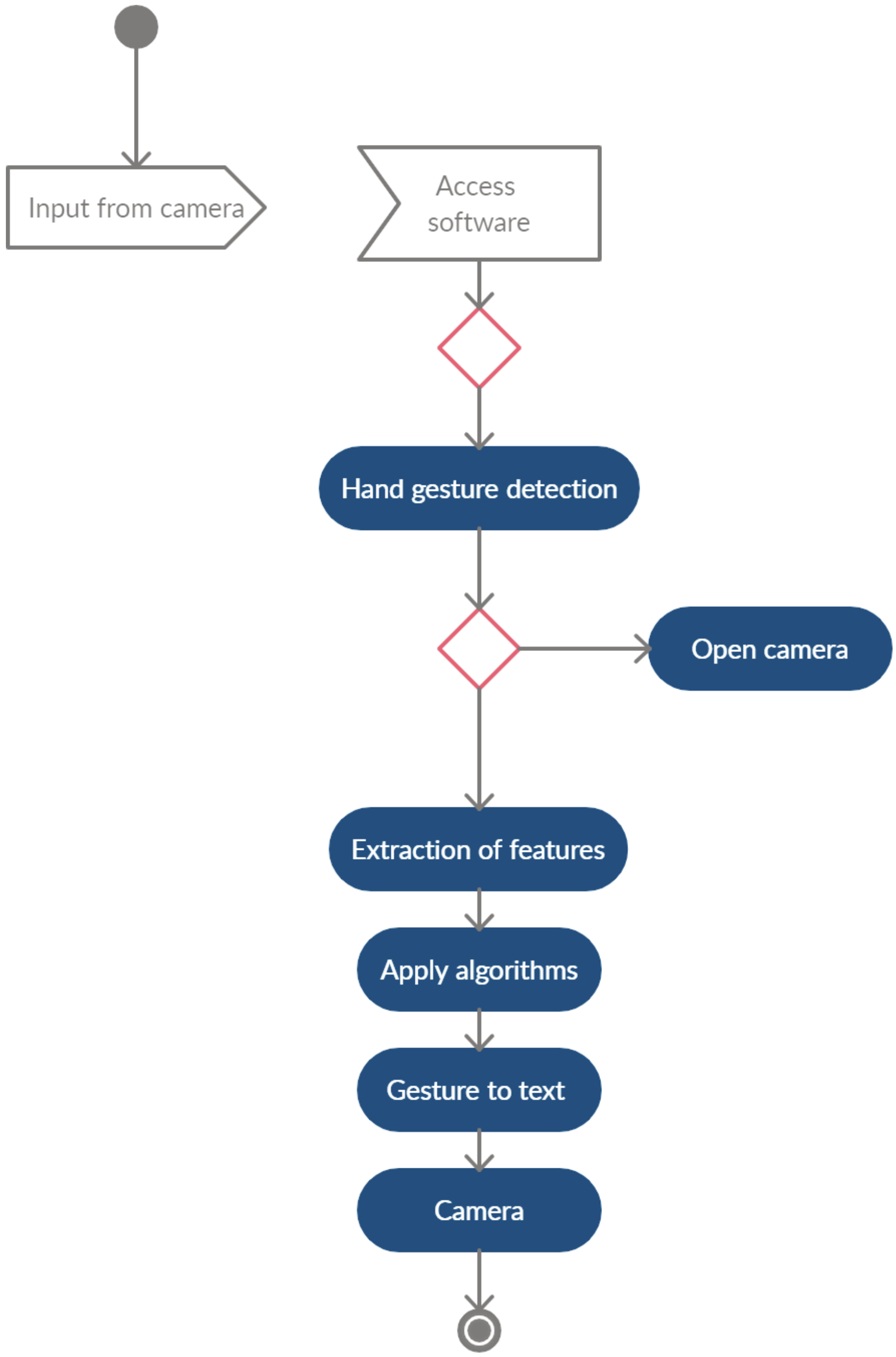

FIG 4.4 Activity diagram of hand gesture recognition system

4.5 Class Diagram

The class diagram is the fundamental component of object-oriented modeling. It is used for both general conceptual modeling of the application's structure and precise modeling of the models' translation into programming code.

So, there are six separate classes in the Sign Language Illustrator class diagram, and a connection between them is shown. The user class is called first, and it calls the detect class, which is the main class, which then calls the initialize webcam class, which captures the image. After capturing, the images need to be passed the image to the training phase so next the train model class will be call and then after it recognize gesture class will be invoked which will detect the gesture and after detecting the gesture the output of the gesture will be print so print text class will be call at the end.

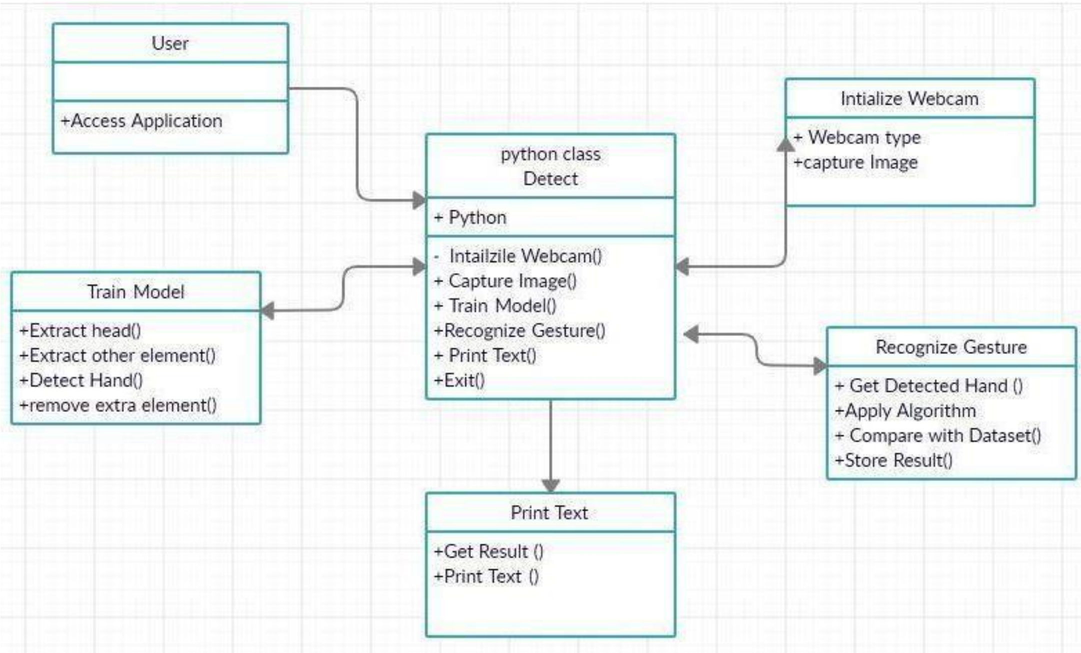

FIG 4.5 Class Diagram of hand gesture recognition system

4.6 Sequence Diagram

The interaction between objects in a sequence diagram is simply represented in the order in which these interactions occur. A sequence diagram may also be referred to as an event diagram or an event scenario.

In the sequential diagram of sign language Illustrator, a user will capture an image of the hand, which will then be transferred to a system, then the system will perform hand detection and feature extraction on the image using various algorithms, after which feature matching will be performed, and the result will be printed in text format based on the detection.

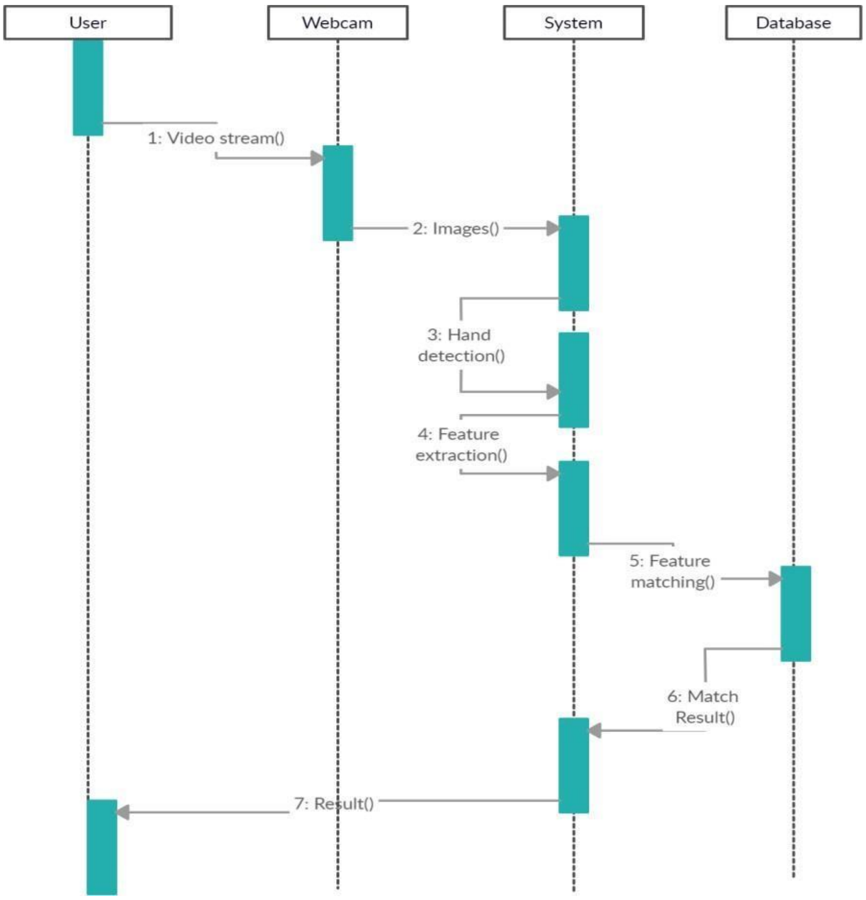

FIG 4.6 Sequence Diagram of hand gesture recognition system

CHAPTER 5

RESULT

We have achieved an accuracy of 93.8% in our model using only layer 1 of our algorithm, and using the combination of layer 1 and layer 2 we achieve an accuracy of 96.0%, which is a better accuracy then most of the current research papers on American sign language. Most of the research papers focus on using devices like Kinect for hand detection. In [7] they build a recognition system for Flemish sign language using convolutional neural networks and Kinect and achieve an error rate of 2.5%. In [8] a recognition model is built using a hidden markov model classifier and a vocabulary of 30words and they achieve an error rate of 10.90%. In [9] they achieve an average accuracy of 86% for 41 static gestures in Japanese sign language. Using depth sensors map [10] achieved an accuracy of 99.99% for observed signers and 83.58% and 85.49% for new signers. They also used CNN for their recognition system. One thing should be noted that our model doesn’t use any background subtraction algorithm while some of the models present above do that. So, once we try to implement background subtraction in our project the accuracies may vary. On the other hand, most of the above projects use Kinect devices but our main aim was to create a project which can be used with readily available resources. A sensor like Kinect not only isn’t readily available but also is expensive for most of audience to buy and our model uses a normal webcam of the laptop hence it is great plus point.

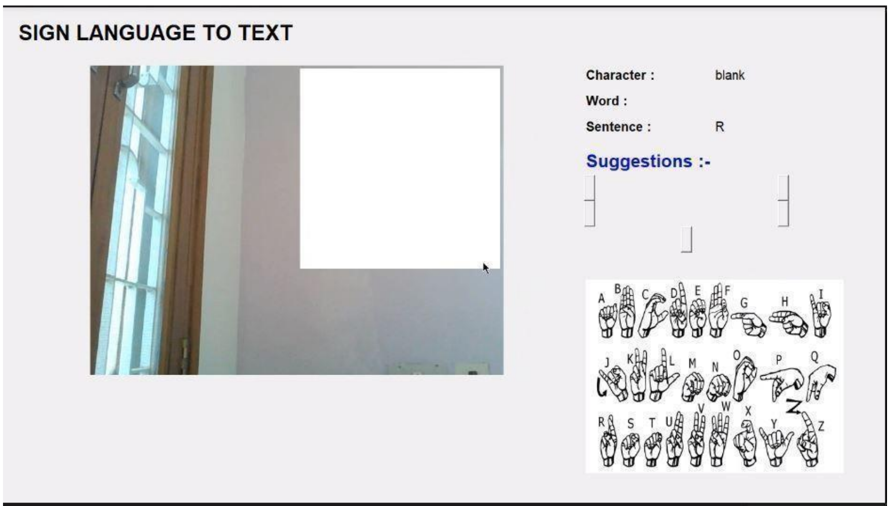

FIG 5.1 Output (Blank)

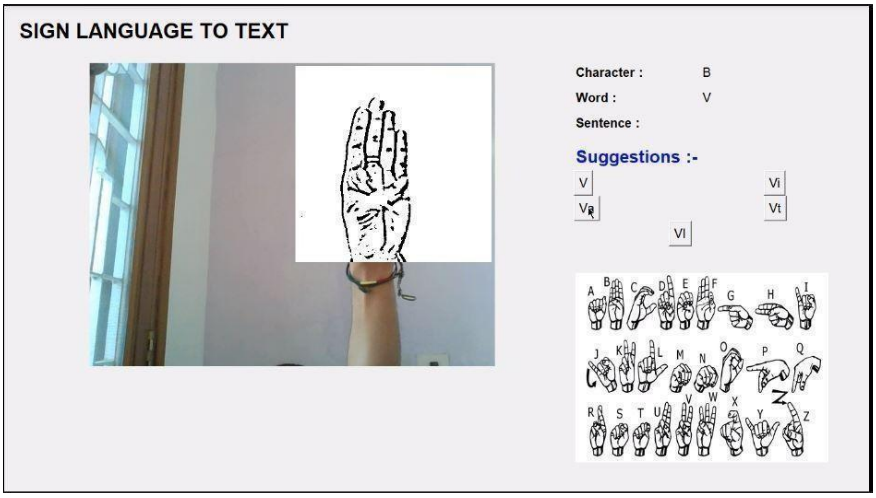

FIG 5.2 Output (Prediction of letter)

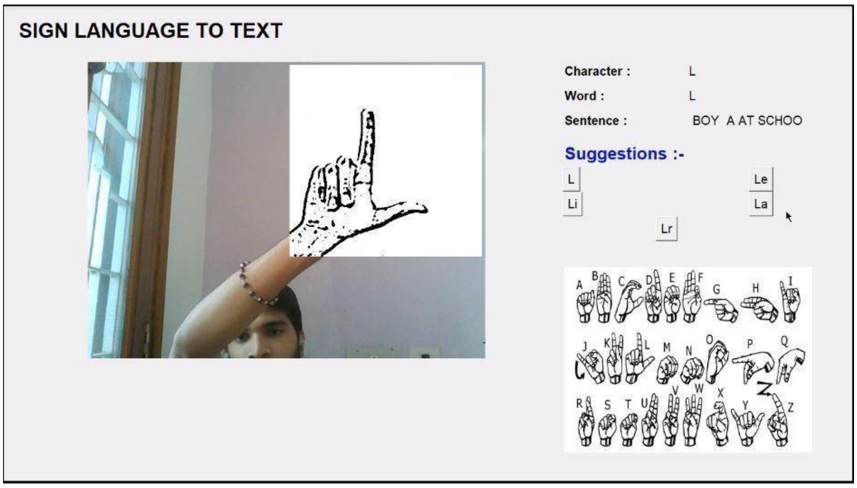

FIG 5.3 Output (sentence formed)

CHAPTER 6

TESTING

We convert our input images (RGB) into grayscale and apply Gaussian blur to remove unnecessary noise. We apply an adaptive threshold to extract our hand from the background and resize our images to 128 x 128.We feed the input images after preprocessing to our model for training and testing after applying all the operations mentioned above. The prediction layer estimates how likely the image will fall under one of the classes. So, the output is normalized between 0 and 1 and such that the sum of each value in each class sums to 1. We have achieved this using the SoftMax function. At first the output of the prediction layer will be somewhat far from the actual value. To make it better we have trained the networks using labeled data. The cross-entropy is a performance measurement used in the classification. It is a continuous function which is positive at values which is not the same as labeled value and is zero exactly when it is equal to the labeled value. Therefore, we optimized the cross-entropy by minimizing it as close to zero. To do this in our network layer we adjust the weights of our neural networks. TensorFlow has an inbuilt function to calculate the cross entropy. As we have found out about the cross-entropy function, we have optimized it using Gradient Descent in fact the best gradient descent optimizer is called Adam Optimizer.

CHAPTER 7

CONCLUSION AND FUTURE SCOPE

7.1 Conclusion

In this report, a functional real time vision based American sign language recognition for D&M people have been developed for asl alphabets. We achieved final accuracy of 96.0% on our dataset. We are able to improve our prediction after implementing two layers of algorithms in which we verify and predict symbols which are more similar to each other. This way we are able to detect almost all the symbols provided that they are shown properly, there is no noise in the background and lighting is adequate.

7.2 Future Scope:

We are planning to achieve higher accuracy even in case of complex backgrounds by trying out various background subtraction algorithms. We are also thinking of improving the preprocessing to predict gestures in low light conditions with a higher accuracy.

References

[1] T. Yang, Y. Xu, and “A., Hidden Markov Model for Gesture Recognition”, CMU-RI-TR-94 10, Robotics Institute, Carnegie-Mellon Univ., Pittsburgh, PA, May 1994.

[2] Pujan Ziaie, Thomas M uller, Mary Ellen Foster, and Alois Knoll “A Naive Bayes Munich, Dept. of Informatics VI, Robotics and Embedded Systems, Boltzmannstr. 3, DE-85748 Garching, Germany.

[3]https://docs.opencv.org/2.4/doc/tutorials/imgproc/gausian_median_blur_bilateral_filter/gausian_median_blur_bilateral_filter.html

[4] Mohammed Waleed Kalous, Machine recognition of Auslan signs using PowerGloves: Towards large-lexicon recognition of sign language.

[5] aeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks-Part-2/

[6] http://www-i6.informatik.rwth-aachen.de/~dreuw/database.php

[7] Pigou L., Dieleman S., Kindermans PJ., Schrauwen B. (2015) Sign Language Recognition Using Convolutional Neural Networks. In: AgapitoL., Bronstein M., Rother C. (eds) Computer Vision - ECCV 2014Workshops. ECCV 2014. Lecture Notes in Computer Science, vol 8925.Springer, Cham

[8] Zaki, M.M., Shaheen, S.I.: Sign language recognition using a combination of new vision based features. Pattern Recognition Letters 32(4),572–577 (2011)25[9] N. Mukai, N. Harada and Y. Chang, "Japanese Finger spelling Recognition Based on Classification Tree and Machine Learning," 2017Nicograph International (NicoInt), Kyoto, Japan, 2017, pp. 19-24.doi:10.1109/NICOInt.2017.9

[10] Byeongkeun Kang, Subarna Tripathi, Truong Q. Nguyen ” Real-time sign language fingerspelling recognition using convolutional neural networks from depth map” 2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)

[11]https://opencv.org/

[12]https://en.wikipedia.org/wiki/TensorFlow

[13]https://en.wikipedia.org/wiki/Convolutional_neural_network

APPENDICES

OpenCV

OpenCV (Open Source Computer Vision Library) is released under a BSD license and hence it's free for both academic and commercial use. It has C++, Python and Java interfaces and supports Windows, Linux, Mac OS, iOS and Android. OpenCV was designed for computational efficiency and with a strong focus on real-time applications. Written in optimized C/C++, the library can take advantage of multi-core processing. Enabled with OpenCL, it can take advantage of the hardware acceleration of the underlying heterogeneous compute platform. Adopted all around the world, OpenCV has more than 47 thousand people of user community and estimated number of downloads exceeding 14 million. Usage ranges from interactive art, to mines inspection, stitching maps on the web or through advanced robotics.

Convolution Neural network

CNNs use a variation of multilayer perceptrons designed to require minimal preprocessing. They are also known as shift invariant or space invariant artificial neural networks (SIANN), based on their shared-weights architecture and translation invariance characteristics. Convolutional networks were inspired by biological processes in that the connectivity pattern between neurons resembles the organization of the animal visual cortex. Individual cortical neurons respond to stimuli only in a restricted region of the visual field known as the receptive field. The receptive fields of different neurons partially overlap such that they cover the entire visual field.

CNNs use relatively little pre-processing compared to other image classification algorithms. This means that the network learns the filters that in traditional algorithms were hand-engineered. This independence from prior knowledge and human effort in feature design is a major advantage. They have applications in image and video recognition, recommended systems, image classification, medical image analysis, and natural language processing.

Tensorflow

TensorFlow is an open-source software library for dataflow programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and production at Google. TensorFlow was developed by the Google brain team for internal Google use. It was released under the Apache 2.0 open source library on November 9, 2015. TensorFlow is Google Brain's second-generation system. Version 1.0.0 was released on February 11, 2017. While the reference implementation runs on single devices, TensorFlow can run on multiple CPUs and GPUs (with optional CUDA and SYCL extensions for general-purpose computing on graphics processing units). TensorFlow is available on 64-bit Linux, macOS, Windows, and mobile computing platforms including Android and iOS. Its flexible architecture allows for the easy deployment of computation across a variety of platforms (CPUs, GPUs, TPUs), and from desktops to clusters of servers to mobile and edge devices.

PUBLISHED PAPERS

1. Our Literature Review Paper has been published in “Journal of University of Shanghai for Science and Technology”, Volume 22, Issue 10-2020 -S.No 54, Manuscript ID: JUSST/10- 218.

2. Our Implementation Paper has been accepted by “RSRI CRSE 2021:4th RSRI Conference on Recent trends in Science and Engineering by REST Labs” and will be published in Scopus Indexed Journal.