VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama, Belagavi – 590018, Karnataka, India

A PROJECT REPORT

On

"Sign Language Interpreter using Deep Learning"

A Dissertation Submitted in partial fulfilment of the requirement for the degree of

BACHELOR OF ENGINEERING

In

COMPUTER SCIENCE & ENGINEERING

Submitted by

Bahaduri Prachiti Jagdish (1RG17CS009)

Mohammed Azim (1RG17CS030)

Mohammed Touseef (1RG17CS031)

Prajwal B Mani (1RG17CS037)

Under The Guidance of

Mrs. Pushplata Dubey

Asst Professor, Dept of CSE

RGIT, Bengaluru-32

Department of Computer Science & Engineering

RAJIV GANDHI INSTITUTE OF TECHNOLOGY

Chola nagar, R.T. Nagar Post, Bengaluru-560032

2020 – 2021

RAJIV GANDHI INSTITUTE OF TECHNOLOGY

(Affiliated to Visvesvaraya Technological University)
Chola nagar, R.T. Nagar Post, Bengaluru-560032

Department of Computer Science & Engineering

CERTIFICATE

Phase – II

This is to certify that the Project Report titled **"Sign Language Interpreter using Deep Learning"** is a bonafide work carried out by **Ms. Bahaduri Prachiti Jagdish (USN 1RG17CS009), Mr. Mohammed Azim (USN 1RG17CS030), Mr. Mohammed Touseef (USN 1RG17CS031) and Mr. Prajwal B Mani (USN 1RG17CS037)** in partial fulfillment for the award of **Bachelor of Engineering** in **Computer Science and Engineering** under **Visvesvaraya Technological University, Belagavi**, during the year **2020 – 2021.** It is certified that all corrections/suggestions given for Internal Assessment have been incorporated in the report. This project report has been approved as it satisfies the academic requirements in respect of project work Phase – II (17CSP85) prescribed for the said degree.

Signature of Guide	Signature of HOD	Signature of Prinicpal
Mrs. Pushplata Dubey	**Mrs. Arudra A**	**Dr. D G Anand**
Asst. Professor	Assoc. Professor & HOD	Principal
Dept. of CSE	Dept. of CSE	RGIT, Bengaluru
RGIT, Bengaluru	RGIT, Bengaluru	

Internal Evaluation

Name of the Examiners	**Signature with Date**
1.	
2.	

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama, Belagavi 590018

RAJIV GANDHI INSTITUTE OF TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

DECLARATION

We hereby declare that the project work entitled **"Sign Language Interpreter using Deep Learning"** submitted to the **Visvesvaraya Technological University, Belagavi** during the academic year **2020 – 2021**, is record of an original work done by us under the guidance of **Mrs. Pushplata Dubey,** Asst. Professor, Department of Computer Science and Engineering, RGIT, Bengaluru in the partial fulfilment of requirements for the award of the degree of **Bachelor of Engineering** in **Computer Science & Engineering.** The results embodied in this project have not been submitted to any other University or Institute for award of any degree or diploma.

Bahaduri Prachiti Jagdish (1RG17CS009)

Mohammed Azim (1RG17CS030)

Mohammed Touseef (1RG17CS031)

Prajwal B Mani (1RG17CS037)

ACKNOWLEDGEMENT

We take this opportunity to thank our college **Rajiv Gandhi Institute of Technology, Bengaluru** for providing us with an opportunity to carry out this project work.

We express our gratitude to **Dr. D G Anand,** Principal, RGIT, Bengaluru, for providing the resources and support without which the completion of this project would have been a difficult task.

We extend our sincere thanks to **Mrs. Arudra A**, Associate Professor and Head, Department of Computer Science and Engineering, RGIT, Bengaluru, for being a pillar of support and encouraging us in the face of all adversities.

We would like to acknowledge the through guidance and support extended towards us by **Mrs. Pushplata Dubey**, Assistant Professor, Dept of CSE, RGIT, Bengaluru and project coordinators **Mrs. Geetha Pawar ,** Assistant Professor, Dept of CSE, RGIT, Bengaluru and **Mrs. Rajani Kodagali**, Assistant Professor, Dept of CSE, RGIT, Bengaluru. Their incessant encouragement and valuable technical support have been of immense help. Their guidance gave us the environment to enhance our knowledge and skills and to reach the pinnacle with sheer determination, dedication and hard work.

We also want to extend our thanks to the entire faculty and support staff of the Department of Computer Science and Engineering, RGIT, Bengaluru, who have encouraged us throughout the course of the Bachelor's Degree.

We want to thank our family for always being there with full support and for providing us with a safe haven to conduct and complete our project. We are ever grateful to them for helping us in these stressful times.

Lastly, we want to acknowledge all the helpful insights given to us by all our friends during the course of this project.

Bahaduri Prachiti Jagdish (1RG17CS009)
Mohammed Azim (1RG17CS030)
Mohammed Touseef (1RG17CS031)
Prajwal B Mani (1RG17CS037)

ABSTRACT

Speech Impairment is a disability, which affects an individual's ability to communicate using speech and hearing. This brings about the difficulty for both the sign and non - sign language speakers to communicate with each other. With recent advances in deep learning and computer vision, the focus of our project is to create a vision of an end to end Convolutional Neural Network that will be trained on the ASL(American Sign Language) dataset then modeled on robust architectures like GoogLeNet/MobileNet architecture and deploy it on an android application so that it will have more accessibility and provides an ease of use, thus aiding communication between signers and non-signers. It is a challenging and interesting problem that if solved will bring a leap in social and technological aspects alike.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

INTRODUCTION

Chapter 1

INTRODUCTION

1.1 Introduction:

Interpreting is a translational activity in which one produces a first and final translation on the basis of a one-time exposure to an expression in a source language. The most common two modes of interpreting are simultaneous interpreting, which is done at the time of the exposure to the source language, and consecutive interpreting, which is done at breaks to this exposure. Language interpretation is defined by the International Standards Organization (ISO). It states that rendering a spoken or signed message into another spoken or signed language, preserving the register and meaning of the source language content. To do so we make use of ASL, which is American Sign Language. American Sign Language is a language that uses hand signs, facial expressions, and body postures to communicate ideas. American Sign Language not only connects the people to those who are deaf but also it serves as a membership card into a linguistic subculture of our society that not everyone is privileged to enjoy. The statistics presented by the International Standard Organization (ISO) show that over 5% of the world's population or 466 million people has disabling hearing loss. And it is expected to rise to 900 million by the year 2050 which is double as compared to the current statistics.

A sign language interpreter must accurately convey messages between two different languages. An interpreter is available for both hearing and deaf individuals. The act of interpreting occurs when a hearing person speaks, and an interpreter renders the speaker's meaning into sign language, or other forms used by the deaf party(ies). The interpreting also happens in reverse, when a deaf person signs, an interpreter renders the meaning expressed in the signs into the oral language for the hearing party, which is sometimes referred to as voice interpreting or voicing. The current statistics estimate that the actual people who are benefited from hearing aid, are only 17% of the total population while the remaining 83% of the people are suffering from the need and use of hea

1.1.1 Deep Learning

Deep learning (also known as deep structured learning) is part of a broader family of machine learning methods based on artificial neural networks with representation learning. Learning can be supervised, semi-supervised or unsupervised.

Deep-learning architectures such as deep neural networks, deep belief networks, recurrent neural networks and convolutional neural networks have been applied to fields including computer vision, machine vision, speech recognition, natural language processing, audio recognition, social network filtering, machine translation, bioinformatics, drug design, medical image analysis, material inspection and board game programs, where they have produced results comparable to and in some cases surpassing human expert performance.

Artificial neural networks (ANNs) were inspired by information processing and distributed communication nodes in biological systems. ANNs have various differences from biological brains. Specifically, neural networks tend to be static and symbolic, while the biological brain of most living organisms is dynamic (plastic) and analog.

1.2 Scope

ASL interpreters work in a large variety of environments, including medical, legal, educational, mental health, vocational, and other environments. Interpreting is often viewed as a practice profession which requires careful judgement of interpersonal and environmental factors as well as expertise in the skills of the profession itself. Degree programs in ASL Interpreting are available at colleges, universities, and technical schools across the country, ranging from associate degrees to master's degrees. In addition, interpreters work with mentors, attend workshops, and get certifications to become more adept, gain experience, and open additional career opportunities. Other interpreters are Children Of Deaf Adults (CODAs) and are typically exposed to ASL and Deaf culture at a young age, giving them an advantage over later learners. In recent years, much research has gone into discerning whether ASL interpreters have access to adequate, specialized, real-world training and career support systems to ensure success and protect against interpreter burnout. Many studies note interpreter reports of frustration with training that proved inadequate to deal with real-world problems, and a lack of professional support. To prevent interpreter burnouts, there are many alternatives ranging from the same human intervention process which costs 100 dollars per hour to interpreting hand gloves which cost 40,000 dollars per pair.

1.3 Motivation

Communication is one of the basic requirements for survival in society. Deaf and dumb people communicate among themselves using sign language but normal people find it difficult to understand their language. Extensive work has been done on American sign language recognition but Indian sign language differs significantly from American sign language.ISL uses two hands for communicating (20 out of 26) whereas ASL uses single hand for communicating. Using both hands often leads to obscurity of features due to overlapping of hands. In addition to this, lack of datasets along with variance in sign language with locality has resulted in restrained efforts in ISL gesture detection. Our project aims at taking the basic step in bridging the communication gap between normal people and deaf and dumb people using Indian sign language. Effective extension of this project to words and common expressions may not only make the deaf and dumb people communicate faster and easier with outer world, but also provide a boost in developing autonomous systems for understanding and aiding them.

1.4 Problem Identification

According to the projection of data it is expected to rise to 900 million by 2050 that is double the current stats. Over 5% of the world's population which is almost 466 million people has disabling hearing loss. Studies reveal that deaf people are around twice as likely to suffer from psychological problems such as depression and anxiety. Hearing loss can affect a person in three main ways firstly, fewer educational and job opportunities due to impaired communication. Social withdrawal due to reduced access to services and difficulties communicating with others emotional problems caused by a drop in self-esteem and confidence.

1.5 Objectives

1. Create a robust android application that aids in communication for deaf people.
2. Use deep learning with transfer learning techniques to build a neural network model that as to ability to learn pattern in video and classify the images.
3. An error correction model for identifying the pattern mismatch and correcting it for the audio or text input format.
4. To build a model that has higher accuracy with less bias and overfitting problems.
5. To enhance the model to classify at least 20 classes or more.

1.5.1 Existing systems

There has been much research on the sign language domain. Some of them are traditional ones while some of them are advanced mechanics like sensor gloves. Therefore, these conventional mechanisms cannot effectively deal with the ever-evolving technologies.

1. **TRADITIONAL – BASED APPROACH**

   A sign language interpreter is a person trained in translating between a spoken and a signed language. This usually means somone who interprets what is being said and signs it for someone who can't hear, but understands the sign. The interpreter of course will also interpret and speak the words which convey the meaning of whatever the signing person signs, so that the hearing person can "hear" what is being signed.

2. **GLOVE – BASED APPROACH**

   The signers are required to wear a sensor glove or a coloured glove. The task will be simplified during segmentation process by wearing glove.

Disadvantages

- Lack of availability of the interpreter for that moment.
- The gloves based approach is that the signer has to wear the sensor hardware along with the glove during the operation of the system.
- The cost of traditional based approach will vary from 100 to 200 dollars/hrs and for gloves based approach its costs around 40,000 dollars / pair.

Fig. 1.1: Traditional – based Approach

Fig. 1.2: Glove – based Approach

1.5.2 Proposed system

We have introduced a refined approach using advanced deep learning techniques that is flexible to use by any mobile users. In this presentation, we propose an architecture to detect signs. Image pre-processing makes the existing data and input data normalized. The pipeline allows us to automate machine learning workflow. Transfer learnings make it easy to make the models to learn even small details that are hard to capture in a small network. Kivy gives us an advantages overview by giving us a docile structure to compile to any version of the mobile app.

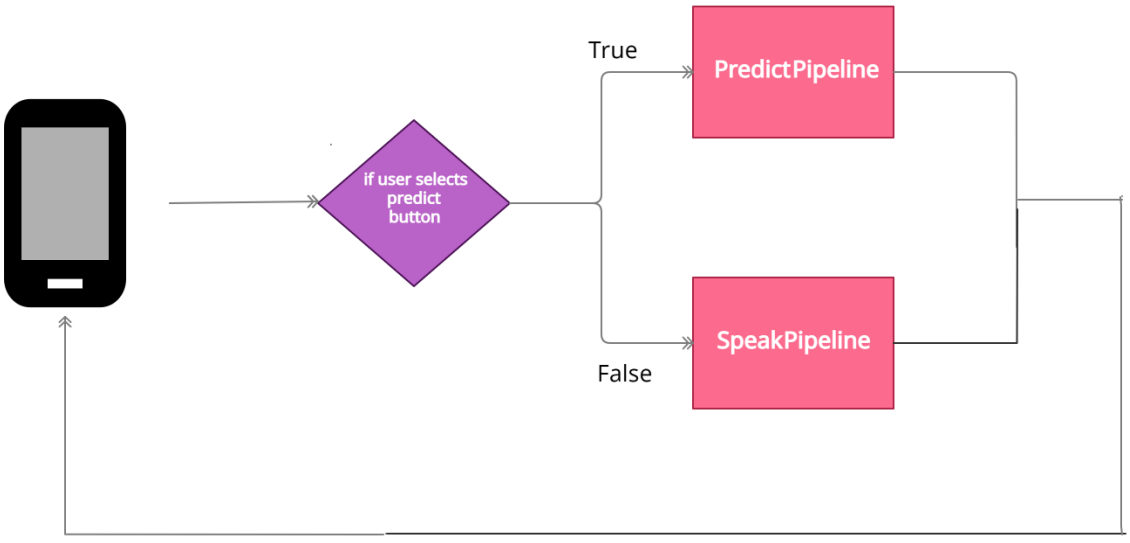


Fig. 1.3: The Architecture of Sign Language Interpreter

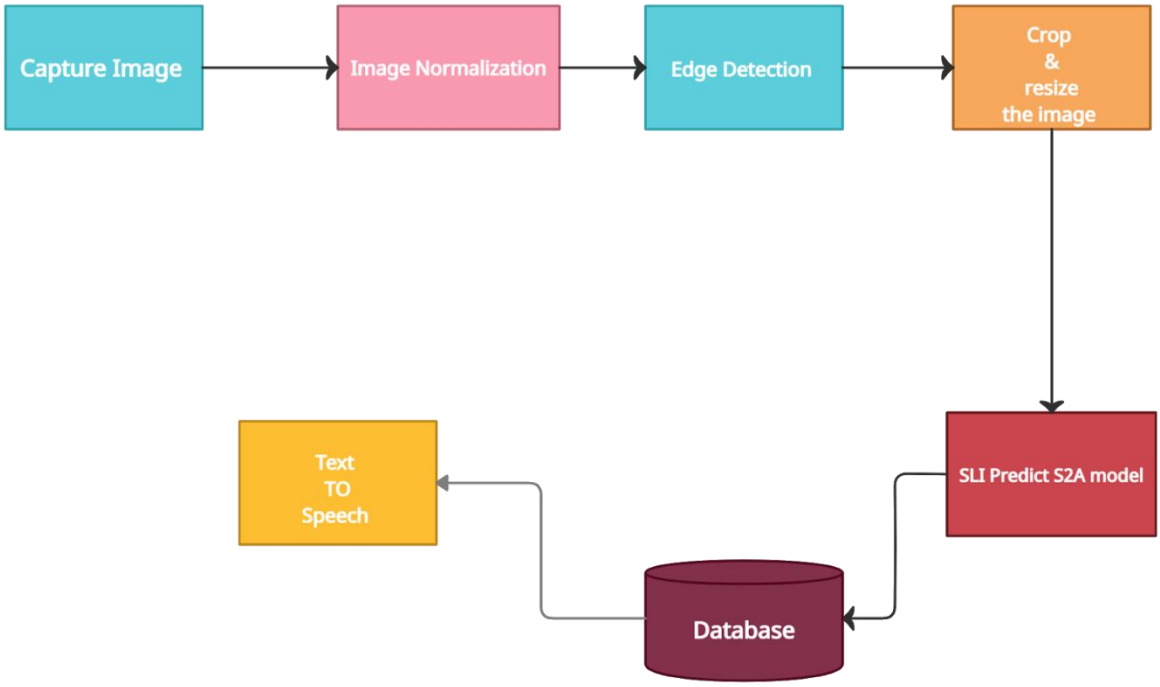


Fig. 1.4: Predict Pipeline Architecture

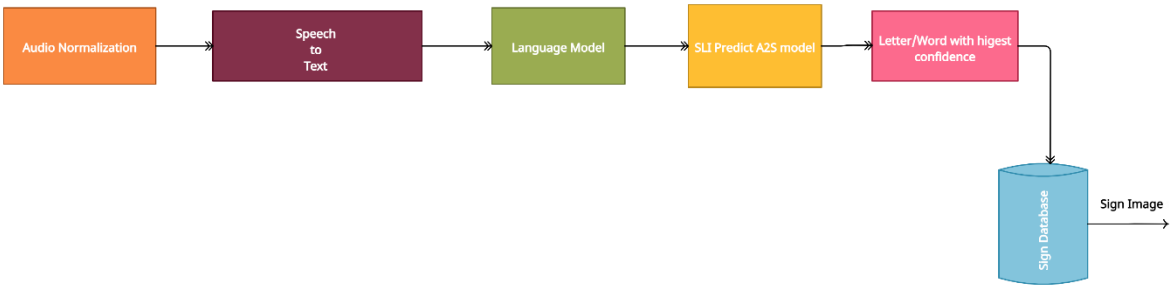


Fig. 1.5: Speak Pipeline Architecture

Advantages

- Considerably better accuracy compared to the previous models.
- Considerably less requirement of raw computation and Storage space requirements.
- Usage of Neural network model leads to less human intervention.

OUTCOME
OF PROJECT

Chapter 2

OUTCOME OF THE PROJECT

Sign Language Interpreter will be able to predict the sign waved by the user and converts that into sound or text vice versa. So far the systems that have been developed give less accuracy and slower results with only shallow networks. The model that has been used in our project will be able to detect with improved accuracy. We will achieve the following results:

1. **UI Interface to access model functionalities:** A flexible UI with a good look and feel with a design that is self-explanatory.
2. **Improved accuracy:** By building a deeper network that model learns and minimizes more parameter that is optimal for our problem
3. **Dynamic Tracing:** By allowing users to dynamical trace would add an advantage when the app is deployed in the real-time
4. **Continuous integration pipelines that enable model redeployment:** Pipelines helps us to automate the machine learning workflows which helps us to reduce time on the surface level.

LITERATURE SURVEY

Chapter 3

LITERATURE SURVEY

A literature survey or a literature review in a project report is that section which shows the various analyses and research made in the field of interest and the results already published, taking into account the various parameters of the project and the extent of the project. It is the most important part of a report as it gives a direction in the area of research. It helps to set a goal for analysis thus giving a problem statement. A literature review is both a summary and explanation of the complete and current state of knowledge on a limited topic as found in academic books and journal articles. Literature survey is a text of a scholarly paper, which includes the current knowledge including substantive findings, as well as theoretical and methodological contributions to a particular topic. And in general, a literature survey guides or helps the researcher to define/find out/identify a problem. The purpose of a literature survey is to: Place each work in the context of its contribution to understanding the research problem being studied. Describes the relationship of each work to the others under consideration. Identify new ways to interpret prior research. Literature of knowledge has a function to teach. It means that literature gives particular values, messages, and themes to the readers. The final function of literature is that literature relieve human either writers or readers from the pressure of emotions. Literature also functions to contribute values of human lives. It provides an excellent starting point for researchers beginning to do research in a new area by forcing them to summarize, evaluate, and compare original research.

As the technology progresses, the internet is now commonly used on PCs, tablets, and smartphones. This generates a huge amount of data, especially textual data. It has become impossible to manually analyze all the data for a specific purpose. New research directions have emerged from automatic data analysis like automatic emotion analysis. Emotion analysis has attracted researchers' attention because of its applications in different fields. Components such as behavior, voice, posture, vocal intensity, and emotion intensity of the a person depicting the emotion, when combined, helps in measuring and recognizing various emotions.

NumPy CNN Android : A Library for Straightforward Implementation of Convolutional Neural Networks for Android Devices

Table 3.1: NumPy CNN Android : A Library for Straightforward Implementation of Convolutional Neural Networks for Android Devices

Name of the paper	Authors	Year	Features	Advantages	Disadvantages
NumPy CNN Android : A Library for Straightforward Implementation of Convolutional Neural Networks for Android Devices	Ahmed Fawzy Gad	2019	AlexNet , VGGNet a nd GoogLeNet are examples of CNN architecture s trained with the ImageNet dataset.	The process of creating a CNN model working on mobile devices is straight forward as the trained model is compatible with them without any in-between transformation steps. Mean error across the samples used is 0.5113.	The weakness of the proposed library is its computational time.

Authors : Ahmed Fawzy Gad

A new open source library called NumPyCNNAndroid is proposed that minimizes the overhead of building and running convolutional neural networks on Android devices. The library is written in Python 3. It uses Kivy for building the application interface and Numerical Python for building the network itself. The library supports the most common layers. Compared to the widely known deep learning libraries, NumPyCNNAndroid avoids the extra overhead of making the network suitable for running on mobile devices. The experimental results validate the correctness of the library implementation by comparing results from both the proposed library and TensorFlow based on mean absolute error.

Deep Learning for American Sign Language Fingerspelling Recognition System

Table 3.2: Deep Learning for American Sign Language Fingerspelling Recognition System

Name of the paper	Authors	Year	Features	Advantages	Disadvantages
Deep Learning for American Sign Language Fingerspelling Recognition System	Huy B.D Nguyen, Hung Ngoc Do	2019	Histograms of Oriented Gradients(HOG) and Local Binary Pattern(LBP), Moore Neighbourhood algorithm, SIFT algorithm	The combinations of HOG, LBP features and multi-kernel multi-class VM acts as a good standalone Feature extractor, better result can be obtained than using an end-to-end CNN architecture.	The validation accuracy of the CNN-SVM model is lower than that of HOG-LBP-SVM model, it has a better chance to counter overfitting.

Authors - Huy B.D Nguyen, Hung Ngoc Do

Sign language has always been a major tool for communication among people with disabilities. In this paper, a sign language fingerspelling alphabet identification system would be developed by using image processing technique, supervised machine learning and deep learning. In particular, 24 alphabetical symbols are presented by several combinations of static gestures (excluding 2 motion gestures J and Z). Histogram of Oriented Gradients (HOG) and Local Binary Pattern (LBP) features of each gesture will be extracted from training images. Then Multiclass Support Vector Machines (SVMs) will be applied to train these extracted data. Also, an end-to-end Convolutional Neural Network (CNN) architecture will be applied to the training dataset for comparison. After that, a further combination of CNN as feature descriptor and SVM produces an acceptable result. The Massey Dataset is implemented in the training and testing phases of the whole system.

American Sign Language Video Hand Gestures Recognition using Deep Neural Networks

Table 3.3: American Sign Language Video Hand Gestures Recognition using Deep Neural Networks

Name of the paper	Authors	Year	Features	Advantages	Disadvantages
American Sign Language Video Hand Gestures Recognition using Deep Neural Networks	Shivashankara S & Srinath S	2019	Speeded Up Robust Features(SURF), Zernike Moment(ZM), Discrete Cosine Transform(DCF), Radon Features	Due to advance preprocessing the PVGR has an average accuracy of 96.43 Overall, the recognition rate obtained is better comparing with the state of art techniques.	It is noticed that, due to the gestures captured in low illumination night time, there is a bit of loss of recognition rate.

Authors : Shivashankara S & Srinath S

In this paper an effort has been placed to translate / recognize some of the video based hand gestures of American Sign Language (ASL) into human and / or machine readable English text using deep neural networks. Initially, the recognition process is carried out by fetching the input video gestures. In the recognition process of the proposed algorithm, for background elimination and foreground detection, the Gaussian Mixture Model (GMM) is used. The basic preprocessing operations are used for better segmentation of the video gestures. The various feature extraction techniques like, Speeded Up Robust Features (SURF), Zernike Moment (ZM), Discrete Cosine Transform (DCT), Radon Features (RF), and R, G, B levels are used to extract the hand features from frames of the video gestures. The extracted video hand gesture features are used for classification and recognition process in forthcoming stage. For classification and followed by recognition, the Deep Neural Networks (stacked autoencoder) is used. This video hand gesture recognition system can be used as tool for filling the communication gap between the normal and hearing impaired people. As a result of this proposed ASL video hand gesture recognition (VHGR), an average recognition rate of 96.43% is achieved. This is the better and motivational performance compared to state of art techniques.

American Sign Language Recognition using Deep Learning and Computer Vision

Table 3.4: American Sign Language Recognition using Deep Learning and Computer Vision

Name of the paper	Authors	Year	Features	Advantages	Disadvantages
American Sign Language Recognition using Deep Learning and Computer Vision	Kshitij Bantupalli & Ying Xie	2018	Use of Inception net, a CNN for recognizing spatial features. then use a RNN to train on temporal features. The dataset used is the American Sign Language Dataset.	Softmax layer gives more accuracy than the pool layer	One of the problems the model faced is with facial features and skin tones. While testing with different skin tones, the model dropped accuracy if it hadn't been trained on a certain skin tone and was made to predict it.

Authors : Kshitij Bantupalli & Ying Xie

Speech impairment is a disability which affects an individuals ability to communicate using speech and hearing. People who are affected by this use other media of communication such as sign language. Although sign language is ubiquitous in recent times, there remains a challenge for non-sign language speakers to communicate with sign language speakers or signers. With recent advances in deep learning and computer vision there has been promising progress in the fields of motion and gesture recognition using deep learning and computer vision-based techniques. The focus of this work is to create a vision- based application which offers sign language translation to text thus aiding communication between signers and non-signers. The proposed model takes video sequences and extracts temporal and spatial features from them. We then use Inception, a CNN (Convolutional Neural Network) for recognizing spatial features. We then use a RNN (Recurrent Neural Network) to train on temporal features. The dataset used is the American Sign Language Dataset.

American Sign Language Character Recognition using Convolutional Neural Network

Table 3.5: American Sign Language Character Recognition using Convolutional Neural Network

Name of the paper	Authors	Year	Features	Advantages	Disadvantages
American Sign Language Character Recognition using Convolutional Neural Network	Sarfaraz Masood, Manish Chandra Thuwal, Adhyan Srivastava	2018	Convolutional Neural Network Model, Image Preprocessing and VGG16 Model	The Convolution Neural Network provides a remarkable accuracy in identifying the sign language characters including alphabets and numerals.	Even though the accuracy is high but the percentage of misclassification is more than expected

Authors : Sarfaraz Masood, Manish Chandra Thuwal, Adhyan Srivastava

Inability to speak is considered to be a true disability. People with this disability use different modes to communicate with others, there are number of methods available for their communication one such common method of communication is sign language. Developing sign language application for deaf people can be very important, as they‟ll be able to communicate easily with even those who don‟t understand sign language. This work aims at taking the basic step in bridging the communication gap between normal people and deaf and dumb people using sign language. The image dataset consists of 2524 ASL gestures. The accuracy of the model obtained using Convolution Neural Network was 96%.

Optimization of Transfer Learning for Sign Language Recognition Targeting Mobile Platform

Table 3.6: Optimization of Transfer Learning for Sign Language Recognition Targeting Mobile Platform

Name of the paper	Authors	Year	Features	Advantages	Disadvantages
Optimization of Transfer Learning for Sign Language Recognition Targeting Mobile Platform	Dhruv Rathi	2018	Convolutional Neural Network(CNN) on ImageNet, two pre-trained models, Inception V3 model and MobileNets open-source models for comparison purposes.	Both mobile net and inception gives us an accuracy more than 95% irrespective of there parameters difference	Model can't predict sentences due to lack of Natural Language Processing(NLP) and other sentences construction techniques.

Authors : Dhruv Rathi

The target of this research is to experiment, iterate and recommend a system that is successful in recognition of American Sign Language(ASL). It is a challenging as well as an interesting problem that if solved will bring a leap in social and technological aspects alike. In this paper, we propose a real-time recognizer of ASL based on a mobile platform, so that it will have more accessibility and provides an ease of use. The technique implemented is Transfer Learning of new data of Hand gestures for alphabets in ASL to be modelled on various pre-trained high- end models and optimize the best model to run on a mobile platform considering the various limitations of the same during optimization. The data used consists of 27,455 images of 24 alphabets of ASL. The optimized model when ran over a memory-efficient mobile application, provides an accuracy of 95.03% of accurate recognition with an average recognition time of 2.42 seconds. This method ensures considerable discrimination in accuracy and recognition time than the previous research.

A Deep Learning Based Video Classification System Using Multimodality Correlation Approach

Table 3.7: A Deep Learning Based Video Classification System Using Multimodality Correlation Approach

Name of the paper	Authors	Year	Features	Advantages	Disadvantages
A Deep Learning Based Video Classification System Using Multimodality Correlation Approach	Jungheon Lee, Youngsan Koh and Jihoon Yang	2017	Image feature vector extraction, Audio feature vector extraction, Normalization of feature vectors	Method of normalizing each modality to a unit vector so that it can be effectively integrated at the integration stage of the multimodality.	Unit vector normalization should be done to get the required accuracy.

Authors : Jungheon Lee, Youngsan Koh and Jihoon Yang

In this paper, we propose a video event classification system that classifies video events using the correlation of images and sounds extracted from one video. The proposed system has better classification performance than other systems using single modality.

Hand Gesture Recognition Using Deep Learning

Table 3.8: Hand Gesture Recognition Using Deep Learning

Name of the paper	Authors	Year	Features	Advantages	Disadvantages
Hand Gesture Recognition Using Deep Learning	Soeb Hussain and Rupal Saxena	2017	Transfer Learning : VGG16 architecture	The accuracy is far better than video based standard network alexnet by scoring 93.09%	The method was made robust by avoiding skin color segmentation, blob detection, skin area cropping and centroid extraction for unidirectional dynamic gestures.

Authors : Soeb Hussain and Rupal Saxena

In order to offer new possibilities to interact with machine and to design more natural and more intuitive interactions with computing machines, our research aims at the automatic interpretation of gestures based on computer vision. In this paper, we propose a technique which commands computer using six static and eight dynamic hand gestures. The three main steps are: hand shape recognition, tracing of detected hand (if dynamic), and converting the data into the required command. Experiments show 93.09% accuracy.

Real-time American Sign Language Recognition with Convolutional Neural Networks

Table 3.9: Real-time American Sign Language Recognition with Convolutional Neural Networks

Name of the paper	Authors	Year	Features	Advantages	Disadvantages
Real-time American Sign Language Recognition with Convolutional Neural Networks	Brandon Garcia & Sigberto Alarcon Viesca (Stanford University Stanford, CA)	2016	A deep learning framework, in order to develop, test, and run our CNN specifically, they used Berkeley Vision and Learning Center's GoogLeNet pre-trained on the 2012 ILSVRC dataset.	Produce effective accuracy with the letters a-e without any image pre-processing	The lack of variation in our datasets, the validation accuracies we observed during training were not directly reproducible upon testing on the web application.

Authors : Brandon Garcia & Sigberto Alarcon Viesca (Stanford University Stanford, CA)

A real-time sign language translator is an important milestone in facilitating communication between the deaf community and the general public. We hereby present the development and implementation of an American Sign Language (ASL) fingerspelling translator based on a convolutional neural network. We utilize a pre-trained GoogLeNet architecture trained on the ILSVRC2012 dataset, as well as the Surrey University and Massey University ASL datasets in order to apply transfer learning this task. We produced a robust model that consistently classifies letters a-e correctly with first-time users and another that correctly classifies letters a-k in a majority of cases. Given the limitations of the datasets and the encouraging results achieved, we are confident that with further research and more data, we can produce a fully generalizable translator for all ASL letters.

Deaf Talk using 3D Animated Sign Language

Table 3.10: Deaf Talk using 3D Animated Sign Language

Name of the paper	Authors	Year	Features	Advantages	Disadvantages
Deaf Talk using 3D Animated Sign Language	Mateen Ahmed, Mujtaba Idrees, Zain ul Abideen, Rafia Mumtaz, Sana Khalique	2016	With Microsoft's Kinect in the market, especially for hands,used the depth sensor of Kinect to recognize around thousand phrases from ASL and this recognition is based on the hidden Markov model (HMM),also focused on Kinect base hand gestures recognition.	The system provides dual mode of communication so it has been categorized into two independent modules with 84% accuracy	This does not solve the communication problem, as natural language speakers don't understand sign language hence there exists a communication gap between these two communities.

Authors : Mateen Ahmed, Mujtaba Idrees, Zain ul Abideen, Rafia Mumtaz, Sana Khalique

This paper describes a neoteric approach to bridge the communication gap between deaf people and normal human beings. In any community there exists such group of disable people who face severe difficulties in communication due to their speech and hearing impairments. Such people use various gestures and symbols to talk and receive their messages and this mode of communication is called sign language. Yet the communication problem doesn't end here, as natural language speakers don't understand sign language resulting in a communication gap. Towards such ends there is a need to develop a system which can act as an interpreter for sign language speakers and a translator for natural language speakers. For this purpose, a software based solution has been developed in this research by exploiting the latest technologies from Microsoft i.e. Kinect for windows V2. The proposed system is dubbed as Deaf Talk, and it acts as a sign language interpreter and translator to provide a dual mode of communication between sign language speakers and natural language speakers. The dual mode of communication has following independent modules (1) Sign/Gesture to speech conversion (2) Speech to sign language conversion. In sign to speech conversion module, the person with speech inhibition has to place himself within Kinect's field of view (FOV) and then performs the sign language gestures. The system receives the performed gestures through Kinect sensor and then comprehends those gestures by comparing them with the trained

gestures already stored in the database. Once the gesture is determined, it is mapped to the keyword corresponding to that gesture. The keywords are then sent to text to speech conversion module, which speaks or plays the sentence for natural language speaker. In contrast to sign to speech conversion, the speech to sign language conversion module translates the spoken language to sign language. In this case, the normal person places himself in the Kinect sensor's FOV and speaks in his native language (English for this case). The system then converts it into text using speech to text API. The keywords are then mapped to their corresponding pre-stored animated gestures and then animations are played on the screen for the spoken sentence. In this way the disable person can visualize the spoken sentence, translated into a 3D animated sign language. The accuracy of Deaf Talk is 87 percent for speech to sign language conversion and 84 percent for sign language to speech conversion.

SYSTEM REQUIREMENTS

Chapter 4

SYSTEM REQUIREMENTS

To be used efficiently, all computer software needs certain hardware components or other software resources to be present on a computer these prerequisites are known as system requirements. System requirements are the configuration that a system must have in order for a hardware or software application to run smoothly and efficiently. Failure to meet these requirements can result in installation problems or performance problems. The former may prevent a device or application from getting installed, whereas the latter may cause a product to malfunction or perform below expectation or even to hang or crash. System requirements are also known as minimum system requirements. System requirements only tell what system must have and what it must allow users to do. The system requirements are of two types:

- Hardware Requirements
- Software Requirements

4.1 Hardware Requirements

The most common set of requirements defined by any operating system or software application is the physical computer resources, also known as hardware.

For Typical Operating System (Windows 10)

For Developement

- Intel Core i3 3rd gen processor or later.
- 8GB RAM & 500GB disk space
- A good GPU supported with CUDNN
- I/O devices
- Any external or inbuilt camera with minimum pixel resolution 200 x 200 (300pi or 1501pi) 4-megapixel cameras and up.

For Deloyment

An Android Smartphone with data connectivity and camera.

4.2 Software Requirements:

The Software requirements deal with defining software resource requirements and prerequisites that need to be installed on a computer to provide optimal functioning of an application.

Front end: Android Studio

Minimum Api level: 19(kitkat)

4.3 Hardware Requirement Specification

Above mentioned requirements are minimum. Practically I shall recommend the following:

1. Minimum 2-GHz processor
2. RAM 4 GB
3. Hard disk available space minimum of 100 GB

- **CUDNN :** The NVIDIA CUDA® Deep Neural Network library (cuDNN) is a GPU-accelerated library of primitives for deep neural networks. cuDNN provides highly tuned implementations for standard routines such as forward and backward convolution, pooling, normalization, and activation layers.
- **Android Smartphone:** A smartphone is a portable device that combines mobile telephone and computing functions into one unit. They are distinguished from feature phones by their stronger hardware capabilities and extensive mobile operating systems, which facilitate wider software, internet (including web browsing over mobile broadband), and multimedia functionality (including music, video, cameras, and gaming), alongside core phone functions such as voice calls and text messaging.
- **Intel® Core™ Processors - The Powerful Processor:**
  Intel Core processors first came to the desktop in mid-2006, replacing the Pentium line that had previously comprised Intel's high-end processors. The Core “i” names are primarily “high level” categorizations that help differentiate processors within a given generation.
- **GPU:** A graphics processing unit (GPU) is a specialized, electronic circuit designed to rapidly manipulate and alter memory to accelerate the creation of images in a frame buffer intended for output to a display device. GPU will help us to train the models faster without using CPU resources.
- **I/O devices**: To view and input data to the application.
- **RAM**: RAM,also known as random access memory is vital for any system to function.
- **Minimum RAM requirements for our system**: 8 GB
- **Camera**: Most importantly you need a camera that can do 30 fps - that means 30 frames per second. Any less than 24 frames per second and your hands will be blurry.
- **HardDisk:** Minimum requirements for disk capacity is 500GB.

4.4 Software Requirement Specification

- **Front End : Android Studio :** Android Studio is the official integrated development environment (IDE) for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development. It is available for download on Windows, macOS and Linux based operating systems or as a subscription-based service in 2020. It is a replacement for the Eclipse Android Development Tools (E-ADT) as the primary IDE for native Android application development.

  The following features are provided in the current stable version:

  - Gradle-based build support
  - Android-specific refactoring and quick fixes
  - Lint tools to catch performance, usability, version compatibility and other problems
  - ProGuard integration and app-signing capabilities
  - Template-based wizards to create common Android designs and components
  - A rich layout editor that allows users to drag-and-drop UI components, option to preview layouts on multiple screen configurations
  - Support for building Android Wear apps
  - Built-in support for Google Cloud Platform, enabling integration with Firebase Cloud Messaging (Earlier 'Google Cloud Messaging') and Google App Engine
  - Android Virtual Device (Emulator) to run and debug apps in the Android studio.

- **Minimum Api level: 19(kitkat)** : The development of Android started in 2003 by Android, Inc., which was purchased by Google in 2005. There were at least two internal releases of the software inside Google and the OHA before the beta version was released The beta was released on November 5, 2007, while the software development kits (SDK) was released on November 12, 2007. Several public beta versions of the SDK were released. These releases were done through software emulation as physical devices did not exist to test the operating system.

Figure 4.1: Android Versions and their release dates

- **Ubuntu : deploying .apk using Android Studio:** You can create a package for android using the python-for-android project. This page explains how to download and use it directly on your own machine (see Packaging with python-for-android) or use the Buildozer tool to automate the entire process. You can also see Packaging your application for the Kivy Launcher to run kivy programs without compiling them. Kivy applications can be released on an Android market such as the Play store, with a few extra steps to create a fully signed APK. The Kivy project includes tools for accessing Android APIs to accomplish vibration, sensor access, texting etc. These, along with information on debugging on the device, are documented at the main Android page.

- **Video & image processing: Open CV, scikit-image:** Processing a video means, performing operations on the video frame by frame. Frames are nothing but just the particular instance of the video in a single point of time. We may have multiple frames even in a single second. Frames can be treated as similar to an image

  1. Adaptive Threshold

  By using this technique we can apply thresholding on small regions of the frame. So the collective value will be different for the whole frame.

2. **Smoothing**

Smoothing a video means removing the sharpness of the video and providing a blurriness to the video. There are various methods for smoothing such as cv2.Gaussianblur(), cv2.medianBlur(), cv2.bilateralFilter(). For our purpose, we are going to use cv2.Gaussianblur().

3. **Edge Detection**

Edge detection is a useful technique to detect the edges of surfaces and objects in the video. Edge detection involves the following steps:

- Noise reduction
- Gradient calculation
- Non-maximum suppression
- Double threshold
- Edge tracking by hysteresis

4. **Bitwise Operations**

Bitwise operations are useful to mask different frames of a video together. Bitwise operations are just like we have studied in the classroom such as AND, OR, NOT, XOR.

Neural Network Modeling:Tensorflow(tf.lite), Keras, Pytorch

- **Tensor flow**

There are multiple changes in TensorFlow 2.0 to make TensorFlow users more productive. TensorFlow 2.0 removes redundant APIs, makes APIs more consistent (Unified RNNs, Unified Optimizers), and better integrates with the Python runtime with Eager execution. Many RFCs have explained the changes that have gone into making TensorFlow 2.0. This guide presents a vision for what development in TensorFlow 2.0 should look like. It's assumed you have some familiarity with TensorFlow 1.x.

- **Keras**

Keras is an API designed for human beings, not machines. Keras follows best practices for reducing cognitive load: it offers consistent & simple APIs, it minimizes the number of user actions required for common use cases, and it provides clear & actionable error messages. It also has extensive documentation and developer guides**.**

- **Exascale machine learning**

  Built on top of TensorFlow 2.0, Keras is an industry-strength framework that can scale to large clusters of GPUs or an entire TPU pod. It's not only possible; it's easy.

- **Deploy anywhere**

  Take advantage of the full deployment capabilities of the TensorFlow platform. You can export Keras models to JavaScript to run directly in the browser, to TF Lite to run on iOS, Android, and embedded devices. It's also easy to serve Keras models as via a web API.

- **Architectures**

  Google LeNet, MobileNet, VGG 16 which are pre-trained models on huge datasets with parameters varying through millions together and deep networks.

  VGG16: It is a convolutional neural network model proposed by K. Simonyan and A. Zisserman from the University of Oxford in the paper "Very Deep Convolutional Networks for Large-Scale Image Recognition". The model achieves 92.7% top-5 test accuracy in ImageNet, which is a dataset of over 14 million images belonging to 1000 classes. It was one of the famous model submitted to ILSVRC-2014. It makes the improvement over AlexNet by replacing large kernel-sized filters (11 and 5 in the first and second convolutional layer, respectively) with multiple 3×3 kernel-sized filters one after another. VGG16 was trained for weeks and was using the NVIDIA Titan Black GPU's.

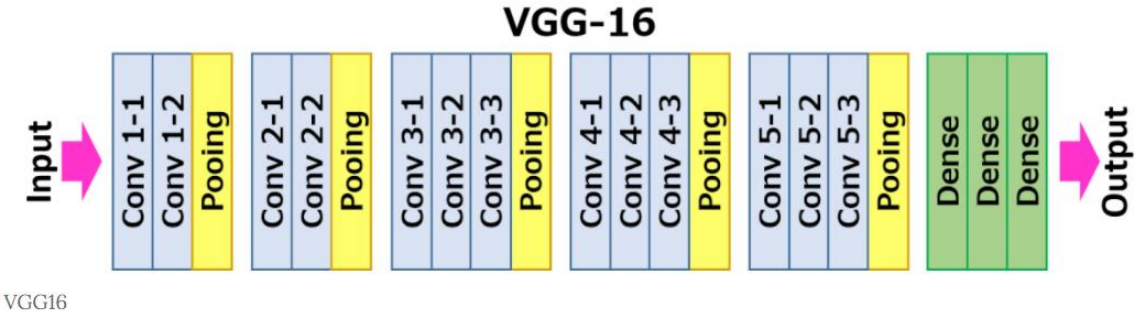


Figure 4.2: VGG – 16 Basic Architecture

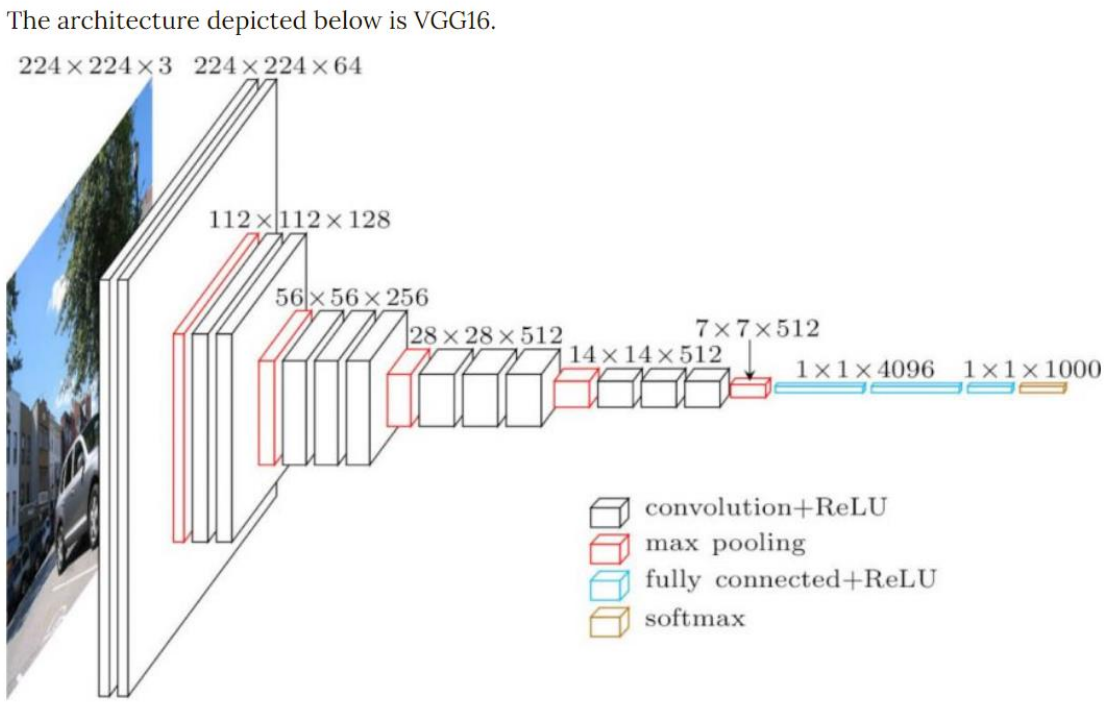


Figure 4.3: VGG – 16 Block Architecture

4.3 System Requirements Summary

This chapter introduces the system analysis process. It gives brief idea whether this project should be done or not based on various feasibility study. It gives the summary of various feasibility studies that were carried out and shows the advantages of doing this project. At the same time, it also gives the overview of various functional and non- functional requirements of the system.

SYSTEM ANALYSIS

Chapter 5

SYSTEM ANALYSIS

5.1 Introduction to System Analysis

- **System:** A system is an orderly group of interdependent components linked together according to a plan to achieve a specific objective. Its main characteristics are organization, interaction, interdependence, integration and a central objective.
- **Analysis:** Analysis is a detailed study of the various operations performed by a system and their relationships within and outside of the system. One aspect of analysis is defining the boundaries of the system and determining whether or not a candidate system should consider other related systems. During analysis data are collected on the available files decision points and transactions handled by the present system. This involves gathering information and using structured tools for analysis.
- **System Analysis:** System analysis and design are the application of the system approach to problem solving generally using computers. To reconstruct a system the analyst must consider its elements output and inputs, processors, controls, feedback and environment.

5.2 Feasibility Study

Feasibility is the determination of whether or not a project is worth doing. The process followed in making this determination is called feasibility Study. This type of study if a project can and should be taken. In the conduct of the feasibility study, the analyst will usually consider seven distinct, but inter-related types of feasibility.

5.2.1 Technical Feasibility

This is considered with specifying equipment and software that will successful satisfy the user requirement the technical needs of the system may vary considerably but might include.

- The facility to provide output at any given time
- Response time under certain condition
- Ability to perform a certain column of transaction at a particular time

5.2.2 Economic Feasibility

Economic analysis is the most frequently used technique for evaluating the effectiveness of a proposed system. More commonly known as cost / benefit analysis. The

procedure is to determine the benefits and savings are expected form a proposed system and a compare them with costs. It benefits outweigh costs; a decision is taken to design and implement the system will have to be made if it is to have a chance of being approved. There is an ongoing effort that improves in accuracy at each phase of the system life cycle.

5.2.3 Operational Feasibility

- What changes will be brought with this system?
- What organizational structures are distributed?
- What new skills will be required?
- Do the existing staff members have the skill?
- If not, can they be trained?

5.3 Functional Requirements

Various functional modules that can be implemented by the system will be:

- **Input Validation:** The system must be able to consider an input from the user to use it for processing.
- **Prediction**: The system must be able to predict the outcome of a test case based on processed results.
- **Data Processing**: The system must be able to process clusters of data in order to form an intelligent system.
- **User Interface**: The system must be able to provide an interface for interaction between user and software.
- **Reporting metrics**: The system should consist of metrics to show performance, accuracy etc to an end user.
- **Distinction**: The system must be able to classify into either blocked or safe divisions.

5.4 Non Functional Requirements

Various non functional modules that can be implemented are

5.4.1 Accuracy

- The system is entirely based of on its ability to be accurate.
- Should show ability to achieve high accuracy on the inputs provided

5.4.2 Speed

- The system must be able to provide results in less time compared to model learning time.
- Speed is essential as input data can vary and can be large.

5.4.3 Security

- The system should not cause any breach such that it causes legal assessment.
- The system must not try to obtain data which might be personal or irrelevant for the task it needs to perform
- The system must not compromise the anonymity of the user accessing its features to other third parties

5.4.4 Consistency

- The system should ensure consistency with its outputs.
- System should be able to provide a consistent and acceptable range of accuracies across its test and validation data.

5.5 System Analysis Summary

This chapter introduces the system analysis process. It gives brief idea whether this project should be done or not based on various feasibility study. It gives the summary of various feasibility studies that were carried out and shows the advantages of doing this project. At the same time, it also gives the overview of various functional and non- functional requirements of the system.

SYSTEM DESIGN

Chapter 6

SYSTEM DESIGN

System design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. Systems design could be seen as the application of systems theory to product development. Object-oriented analysis and design methods are becoming the most widely used methods for computer systems design. The UML has become the standard language in object-oriented analysis and design.

6.1 Logical Design

User interface (UI) design is the process of making interfaces in software or computerized devices with a focus on looks or style. Designers aim to create designs users will find easy to use and pleasurable. UI design typically refers to graphical user interfaces but also includes others, such as voice-controlled ones.

User interfaces are the access points where users interact with designs. Graphical user interfaces (GUIs) are designs' control panels and faces; voice-controlled interfaces involve oral-auditory interaction, while gesture-based interfaces witness users engaging with 3D design spaces via bodily motions. User interface design is a craft that involves building an essential part of the user experience; users are very swift to judge designs on usability and likeability. Designers focus on building interfaces users will find highly usable and efficient. Thus, a thorough understanding of the context's users will find themselves in when making those judgments is crucial.

6.2 Design Goals

The following are the design goals that are applicable to virtually every WebApp regardless of application domain, size, or complexity.

1. Simplicity
2. Consistency
3. Identity
4. Visual appeal
5. Compatibility.

Design leads to a model that contains the appropriate mix of aesthetics, content, and technology. The mix will vary depending upon the nature of the application, and as a consequence the design activities that are emphasized will also vary.

6.3 Data Flow Diagram

A data flow diagram is a graphical representation of the "flow" of data through an information system, modelling its process aspects. A DFD is often used as a preliminary step to create an overview of the system without going into great detail, which can later be elaborated. DFDs can also be used for the visualization of data processing.

A DFD shows what kind of information will be input to and output from the system, how the data will advance through the system, and where the data will be stored.

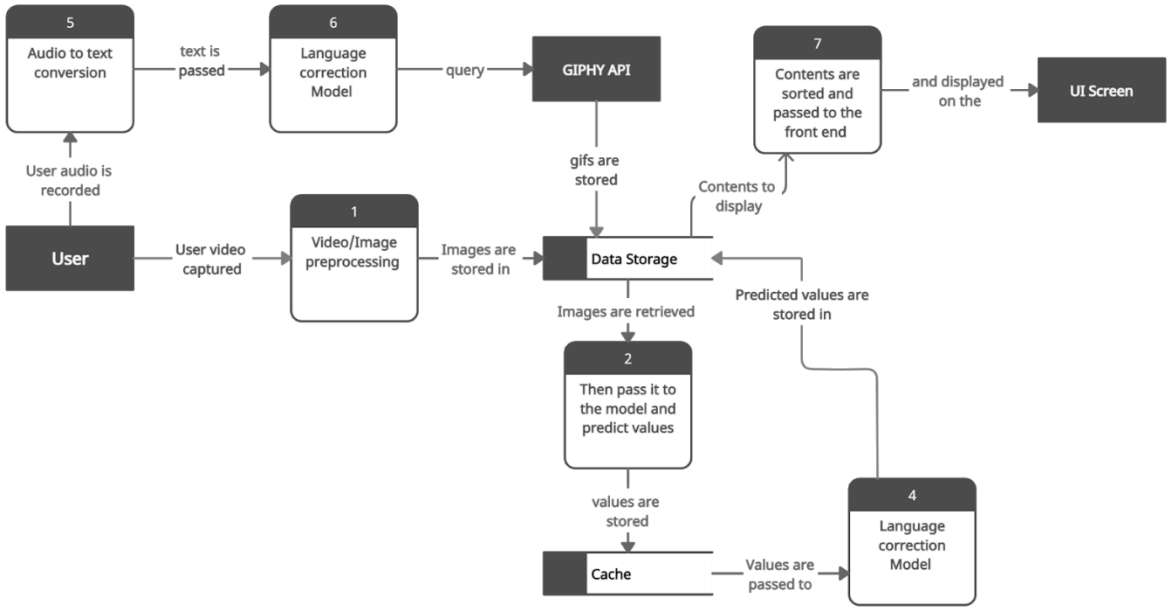


Figure 6.1: Data Flow Diagram

6.4 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. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well.

While a use case itself might drill into a lot of detail about every possibility, a use case diagram can help provide a higher-level view of the system. It has been said before that "Use case diagrams are the blueprints for your system". They provide the simplified and graphical representation of what the system must actually do.

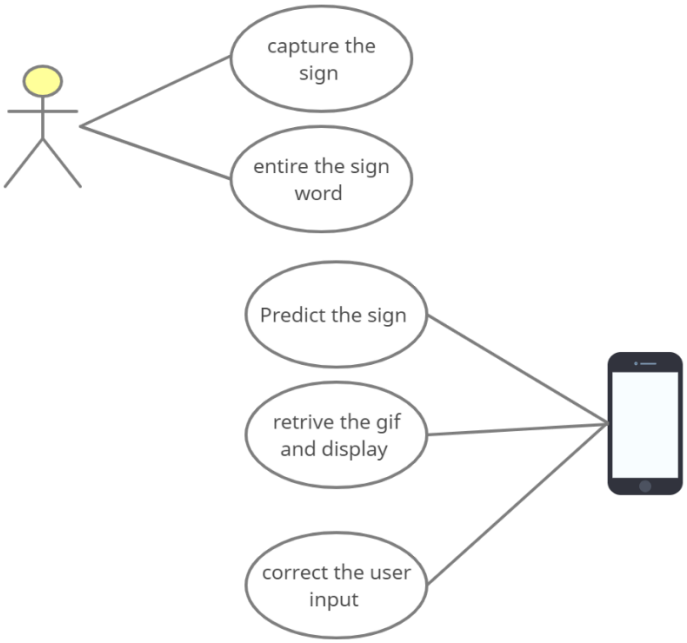


Figure 6.2: Use Case Diagram

6.5 Sequence Diagram

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios.

A sequence diagram shows, as parallel vertical lines (lifelines), different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.

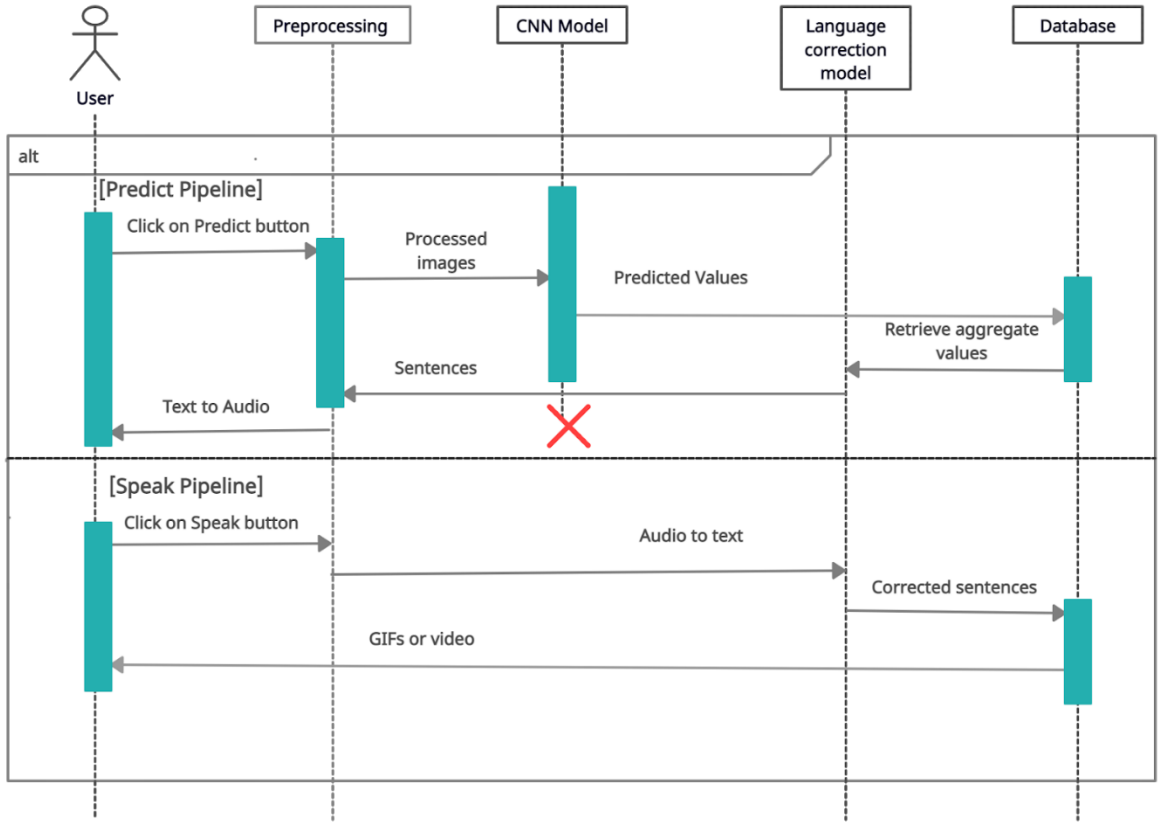


Figure 6.3: Sequence Diagram

6.6 Class Diagram

In software engineering, a class diagram in the Unified Modeling 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 (or methods), and the relationships among objects.

Purpose of Class Diagrams

1. Shows static structure of classifiers in a system
2. Diagram provides basic notation for other structure diagrams prescribed by UML
3. Helpful for developers and other team members too
4. Business Analysts can use class diagrams to model systems from business perspective.

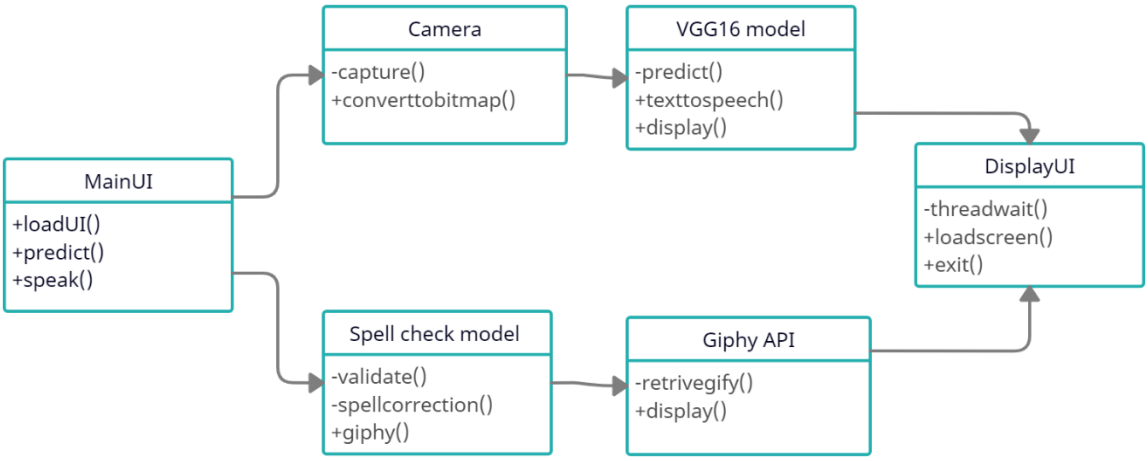


Figure 6.4: Class Diagram

6.7 Life Cycle Model

The agile model is applied for the software development process in our project.

Agile means relating to or denoting a method of project management, that is characterized by the division of tasks into short phases of work and frequent reassessment and adaptation of plans.

There are 5 phases involved in our project:

1. **Requirements**: Recognizing hardware and software requirements
2. **Design**: developing and working on architecture diagram, class diagram, use case design and sequence diagram

3. Development: Training the model using Tensorflow with Transfer Learning and building an UI using Android Studio

4. Testing: Testing with various images and checking whether it classifies the images and signs are retrieved properly

5. Deployment: Deploying on an Android Smartphone using APK feature from Android Studio

6. Review: Test whether accuracy desired has been achieved. Repetitively, this process iterates to improve accuracy of classification.

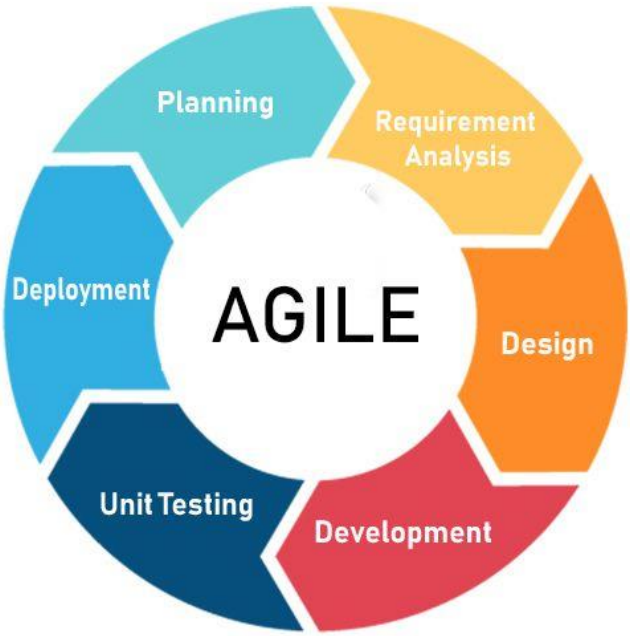


Figure 6.5: Agile Model

6.8 Gantt Chart

6.8.1 The Gantt Chart for Phase One

The Gantt chart has been prepared for first phase of the project. The single date tasks have been depicted as milestones and are shown along the horizontal line. Similarly, the tasks which take multiple days to accomplish are shown in the vertical line. The Gantt chart has been shown below:

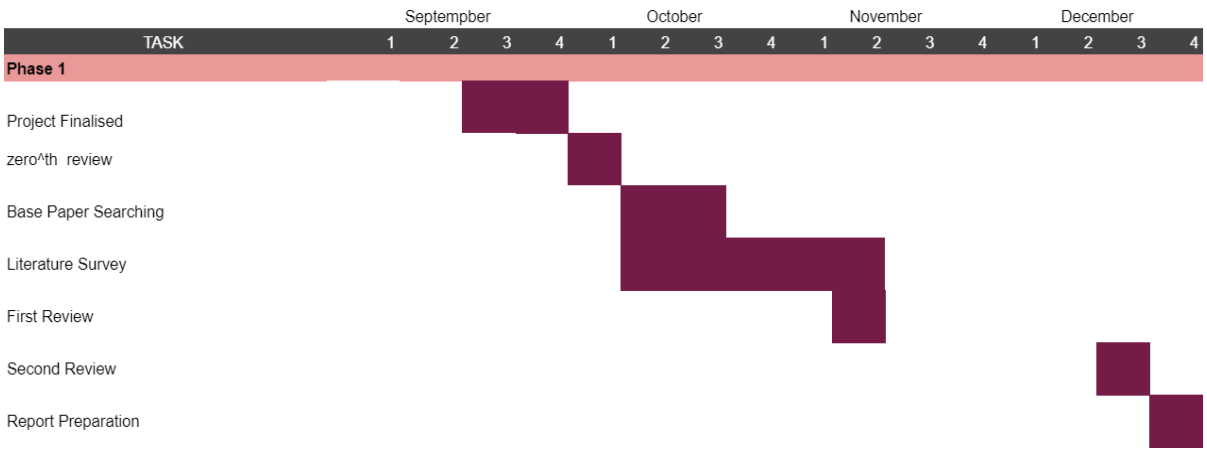


Figure 6.6: Phase one Gantt Chart

6.8.2 The Gantt Chart for Phase Two

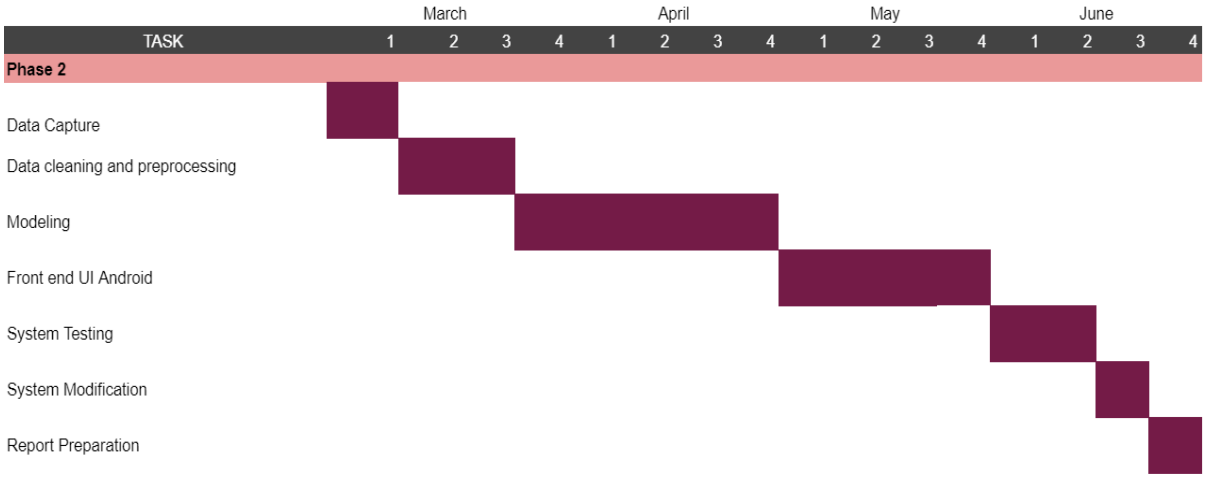


Figure 6.7: Phase two Gantt Chart

6.9 System Design Summary

This chapter shows the detailed design of the architecture, components, modules, interfaces, and data for the proposed system to satisfy specified requirements. It shows various standard UML diagrams that are needed to design the system. It provides a visualization of how the data will flow among various components of the system.

SYSTEM IMPLEMENTATION

Chapter 7

SYSTEM IMPLEMENTATION

Systems implementation is the process of defining how the information system should be built (i.e., physical system design), ensuring that the information system is operational and used, ensuring that the information system meets quality standard (i.e., quality assurance).

System Implementation uses the structure created during architectural design and the results of system analysis to construct system elements that meet the stakeholder requirements and system requirements developed in the early life cycle phases. These system elements are then integrated to form intermediate aggregates and finally the complete system-of-interest (SoI).

Implementation is the process that actually yields the lowest-level system elements in the system hierarchy. System elements are made, bought, or reused. Production involves the hardware fabrication processes of forming, removing, joining, and finishing, the software realization processes of coding and testing, or the operational procedures development processes for operators' roles. If implementation involves a production process, a manufacturing system which uses the established technical and management processes may be required.

The purpose of the implementation process is to design and create a systemelement conforming to that element‘s design properties and/or requirements. The element is constructed employing appropriate technologies and industry practices. This process bridges the system definition processes and the integration process.

7.1 Data Capture

```
import cv2
import uuid
import os
import time

image_path=r'C:\Users\prajw\OneDrive\Desktop\algo\images'
no_of_images=50

while True:
    label=input("enter the label name or q to exit :")
```

```
    if label =='q':
        print("Existing!")
        break
    else:
        os.mkdir(r'C:\Users\prajw\OneDrive\Desktop\algo\images\{}'.format(label))
        c = cv2.VideoCapture(0)
        print("Collecting images foe the label {}".format(label))
        for n in range(no_of_images):
            print("Collecting images for {}".format(n))
            ret, frame = c.read()
            imgname=os.path.join(image_path,label,label+'.'+'{}.jpg'.format(str(uuid.uuid1())))
            cv2.imwrite(imgname, frame)
            cv2.imshow('frame', frame)
            time.sleep(2)
            if cv2.waitKey(1) & 0xFF == ord('q'):
                break
        c.release()

print("Success!")
```

7.2 Crop Images

```
import cv2
import os
import xml.etree.ElementTree as ET

source_dir=input("Enter the source dir path:")
destination_dir=input("Enter the destination dir path:")

for filename in os.listdir(source_dir):
    if filename.endswith('.xml'):
        ymin=0
        ymax=0
        xmin=0
        xmax=0
```

```
img_filename=filename[:-4]+".jpg"
xml_path=os.path.join(source_dir,filename)
tree=ET.parse(xml_path)
xmlroot=tree.getroot()
xmlobject=xmlroot.find("object")
xmlbndbox=xmlobject.find("bndbox")
ymin=int(xmlbndbox.find("ymin").text)
ymax=int(xmlbndbox.find("ymax").text)
xmin=int(xmlbndbox.find("xmin").text)
xmax=int(xmlbndbox.find("xmax").text)
img_path=os.path.join(source_dir,img_filename)
img=cv2.imread(img_path)
try:
    crop_img=img[ymin:ymax,xmin:xmax]
except TypeError:
    continue
dest_img_path=os.path.join(destination_dir,img_filename)
try:
    cv2.imwrite(dest_img_path,crop_img)
except:
    continue
```

7.3 VGG 16 Model Training

In [4]:
```
import numpy as np
import tensorflow as tf
import tensorflow.keras
from tensorflow.keras import utils,callbacks
from tensorflow.keras.models import Sequential,Model
from tensorflow.keras.applications.vgg16 import VGG16
from tensorflow.keras.layers.core import Dropout,Flatten,Dense
from keras.layers.convolutional import Convolution2D,MaxPooling2D,ZeroPadding2D
from keras.optimizers import Adam
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.applications.vgg16 import preprocess_input,decode_predictions
import PIL.Image
from tensorflow import lite
import datetime
```

In [5]:
```
train_dir=r"D:\jupyternotebooks\vgg16\signtrain"
test_dir=r"D:\jupyternotebooks\vgg16\signtest"
```

In [3]:
```
classes=40
batch=512
epochs=10
lnr=0.0001
```

In [4]:
```
img_gen=ImageDataGenerator(rotation_range=20,zoom_range=0.1,
                           width_shift_range=0.1,height_shift_range=0.1,
                           shear_range=0.1,horizontal_flip=True,
                           rescale=1/255.0,validation_split=0.2)

train=img_gen.flow_from_directory(train_dir,target_size=(64,64),subset="training")
eval=img_gen.flow_from_directory(train_dir,target_size=(64,64),subset="validation")
```

```
Found 31190 images belonging to 40 classes.
Found 7789 images belonging to 40 classes.
```

In [6]:
```
adam=Adam(learning_rate=lnr)
model=Sequential()
model.add(VGG16(weights='imagenet',include_top=False,input_shape=(64,64,3)))
model.add(Flatten())
model.add(Dense(256,activation='relu'))
model.add(Dense(40,activation='softmax'))
model.compile(optimizer=adam,loss='categorical_crossentropy',metrics=['accuracy'])
```

In [7]:
```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
vgg16 (Functional)           (None, 2, 2, 512)         14714688
_________________________________________________________________
flatten (Flatten)            (None, 2048)              0
_________________________________________________________________
dense (Dense)                (None, 256)               524544
_________________________________________________________________
dense_1 (Dense)              (None, 40)                10280
=================================================================
Total params: 15,249,512
Trainable params: 15,249,512
Non-trainable params: 0
_________________________________________________________________
```

In [8]:
```
earlystopping=callbacks.EarlyStopping(monitor="val_loss",
                                      mode='min',
                                      patience=5,
                                      restore_best_weights=True)

checkpoint_file_path=r"D:\jupyternotebooks\vgg16\modelcheckpint"
modelcheckpoint= tf.keras.callbacks.ModelCheckpoint(filepath=checkpoint_file_path,
                              monitor="val_loss",
                              verbose=1,
                              save_best_only=True,
                              mode='min'         )
```

In [9]:
```
history=model.fit(train,
                  validation_data=eval,
                  epochs=epochs,
                  shuffle=True,
                  verbose=1,
                  callbacks=[earlystopping,modelcheckpoint])
```

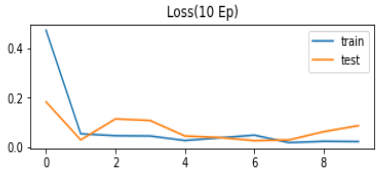
```
Epoch 1/10
975/975 [==============================] - 1118s 1s/step - loss: 1.2061 - accuracy: 0.6659 - val_loss: 0.1831 - val_accuracy: 0.9579

Epoch 00001: val_loss improved from inf to 0.18311, saving model to D:\jupyternotebooks\vgg16\modelcheckpint
INFO:tensorflow:Assets written to: D:\jupyternotebooks\vgg16\modelcheckpint\assets
Epoch 2/10
975/975 [==============================] - 659s 676ms/step - loss: 0.0572 - accuracy: 0.9838 - val_loss: 0.0289 - val_accuracy: 0.9926

Epoch 00002: val_loss improved from 0.18311 to 0.02889, saving model to D:\jupyternotebooks\vgg16\modelcheckpint
INFO:tensorflow:Assets written to: D:\jupyternotebooks\vgg16\modelcheckpint\assets
Epoch 3/10
975/975 [==============================] - 580s 593ms/step - loss: 0.0481 - accuracy: 0.9862 - val_loss: 0.1140 - val_accuracy: 0.9724

Epoch 00003: val_loss did not improve from 0.02889
Epoch 4/10
975/975 [==============================] - 143s 147ms/step - loss: 0.0483 - accuracy: 0.9862 - val_loss: 0.1075 - val_accuracy: 0.9691

Epoch 00004: val_loss did not improve from 0.02889
Epoch 5/10
975/975 [==============================] - 145s 149ms/step - loss: 0.0311 - accuracy: 0.9920 - val_loss: 0.0447 - val_accuracy: 0.9891

Epoch 00005: val_loss did not improve from 0.02889
Epoch 6/10
975/975 [==============================] - 147s 151ms/step - loss: 0.0444 - accuracy: 0.9877 - val_loss: 0.0385 - val_accuracy: 0.9901

Epoch 00006: val_loss did not improve from 0.02889
Epoch 7/10
975/975 [==============================] - 148s 151ms/step - loss: 0.0603 - accuracy: 0.9851 - val_loss: 0.0259 - val_accuracy: 0.9931

Epoch 00007: val_loss improved from 0.02889 to 0.02591, saving model to D:\jupyternotebooks\vgg16\modelcheckpint
INFO:tensorflow:Assets written to: D:\jupyternotebooks\vgg16\modelcheckpint\assets
Epoch 8/10
975/975 [==============================] - 247s 253ms/step - loss: 0.0166 - accuracy: 0.9954 - val_loss: 0.0289 - val_accuracy: 0.9915

Epoch 00008: val_loss did not improve from 0.02591
Epoch 9/10
975/975 [==============================] - 428s 439ms/step - loss: 0.0281 - accuracy: 0.9926 - val_loss: 0.0623 - val_accuracy: 0.9824

Epoch 00009: val_loss did not improve from 0.02591
Epoch 10/10
975/975 [==============================] - 161s 165ms/step - loss: 0.0256 - accuracy: 0.9936 - val_loss: 0.0867 - val_accuracy: 0.9759

Epoch 00010: val_loss did not improve from 0.02591
```

In [11]:
```
model.save(r"D:\jupyternotebooks\vgg16\modelweights")
```

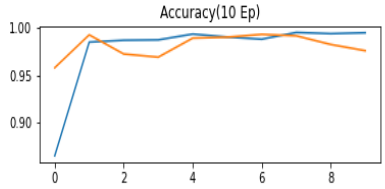
```
INFO:tensorflow:Assets written to: D:\jupyternotebooks\vgg16\modelweights\assets
```

In [16]:
```
model_file="VGG16(10ep).h5"
keras.models.save_model(model,model_file)
```

In [14]:
```
plt.subplot(211)
plt.title('Loss(10 Ep)')
plt.plot(history.history['loss'], label='train')
plt.plot(history.history['val_loss'], label='test')
plt.legend()
plt.savefig('Loss(10 Ep).png')
plt.show()
```



In [15]:
```
plt.subplot(212)
plt.title('Accuracy(10 Ep)')
plt.plot(history.history['accuracy'], label='train')
plt.plot(history.history['val_accuracy'], label='test')
plt.savefig('Accuracy(10 Ep).png')
plt.show()
```



In [33]:
```
# import cv2
# img=cv2.imread(r"D:\jupyternotebooks\vgg16\A.00c43d5e-ae2f-11eb-998f-00f48dddcbd4.jpg")
# resize_img=cv2.resize(img,(64,64))
test=ImageDataGenerator(rescale=1./255).flow_from_directory(test_dir,target_size=(64,64),class_mode=None)
pred=model.predict(test)
pred = np.argmax(pred, axis=1)
pred
```

```
Found 1 images belonging to 1 classes.
```

Out[33]:
```
array([12], dtype=int64)
```

In [22]:
```
adam=Adam(learning_rate=lnr)
model25=Sequential()
model25.add(VGG16(weights='imagenet',include_top=False,input_shape=(64,64,3)))
model25.add(Flatten())
model25.add(Dense(256,activation='relu'))
model25.add(Dense(40,activation='softmax'))
model25.compile(optimizer=adam,loss='categorical_crossentropy',metrics=['accuracy'])
```

In [23]:
```
earlystopping=callbacks.EarlyStopping(monitor="val_loss",
                                      mode='min',
                                      patience=5,
                                      restore_best_weights=True)

checkpoint_file_path=r"D:\jupyternotebooks\vgg16\model50checkpint"
modelcheckpoint= tf.keras.callbacks.ModelCheckpoint(filepath=checkpoint_file_path,
                               monitor="val_loss",
                               verbose=1,
                               save_best_only=True,
                               mode='min'          )
```

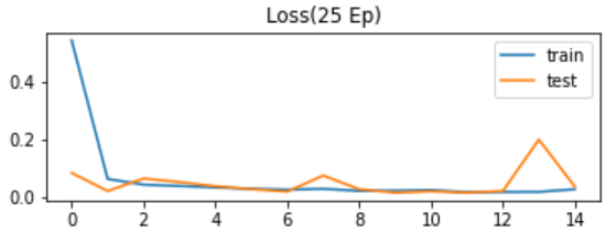
In [24]:
```
history25=model25.fit(train,
                      validation_data=eval,
                      epochs=25,
                      shuffle=True,
                      verbose=1,
                      callbacks=[earlystopping,modelcheckpoint])
```

```
Epoch 1/25
975/975 [==============================] - 1043s 1s/step - loss: 1.3675 - accuracy: 0.6250 - val_loss: 0.0849 - val_accuracy: 0.9737

Epoch 00001: val_loss improved from inf to 0.08486, saving model to D:\jupyternotebooks\vgg16\model50checkpint
INFO:tensorflow:Assets written to: D:\jupyternotebooks\vgg16\model50checkpint\assets
Epoch 2/25
975/975 [==============================] - 188s 193ms/step - loss: 0.0700 - accuracy: 0.9793 - val_loss: 0.0228 - val_accuracy: 0.9936

Epoch 00002: val_loss improved from 0.08486 to 0.02279, saving model to D:\jupyternotebooks\vgg16\model50checkpint
INFO:tensorflow:Assets written to: D:\jupyternotebooks\vgg16\model50checkpint\assets
Epoch 3/25
975/975 [==============================] - 151s 155ms/step - loss: 0.0373 - accuracy: 0.9892 - val_loss: 0.0657 - val_accuracy: 0.9831

Epoch 00003: val_loss did not improve from 0.02279
Epoch 4/25
975/975 [==============================] - 154s 158ms/step - loss: 0.0401 - accuracy: 0.9878 - val_loss: 0.0536 - val_accuracy: 0.9874

Epoch 00004: val_loss did not improve from 0.02279
Epoch 5/25
975/975 [==============================] - 157s 161ms/step - loss: 0.0329 - accuracy: 0.9920 - val_loss: 0.0390 - val_accuracy: 0.9895

Epoch 00005: val_loss did not improve from 0.02279
Epoch 6/25
975/975 [==============================] - 159s 163ms/step - loss: 0.0290 - accuracy: 0.9923 - val_loss: 0.0293 - val_accuracy: 0.9905

Epoch 00006: val_loss did not improve from 0.02279
Epoch 7/25
975/975 [==============================] - 161s 165ms/step - loss: 0.0324 - accuracy: 0.9913 - val_loss: 0.0208 - val_accuracy: 0.9946

Epoch 00007: val_loss improved from 0.02279 to 0.02081, saving model to D:\jupyternotebooks\vgg16\model50checkpint
INFO:tensorflow:Assets written to: D:\jupyternotebooks\vgg16\model50checkpint\assets
Epoch 8/25
975/975 [==============================] - 159s 163ms/step - loss: 0.0186 - accuracy: 0.9953 - val_loss: 0.0759 - val_accuracy: 0.9792


Epoch 00008: val_loss did not improve from 0.02081
Epoch 9/25
975/975 [==============================] - 158s 162ms/step - loss: 0.0226 - accuracy: 0.9944 - val_loss: 0.0289 - val_accuracy: 0.9935

Epoch 00009: val_loss did not improve from 0.02081
Epoch 10/25
975/975 [==============================] - 159s 163ms/step - loss: 0.0178 - accuracy: 0.9956 - val_loss: 0.0169 - val_accuracy: 0.9951

Epoch 00010: val_loss improved from 0.02081 to 0.01693, saving model to D:\jupyternotebooks\vgg16\model50checkpint
INFO:tensorflow:Assets written to: D:\jupyternotebooks\vgg16\model50checkpint\assets
Epoch 11/25
975/975 [==============================] - 160s 164ms/step - loss: 0.0164 - accuracy: 0.9957 - val_loss: 0.0217 - val_accuracy: 0.9946

Epoch 00011: val_loss did not improve from 0.01693
Epoch 12/25
975/975 [==============================] - 159s 163ms/step - loss: 0.0156 - accuracy: 0.9963 - val_loss: 0.0171 - val_accuracy: 0.9963

Epoch 00012: val_loss did not improve from 0.01693
Epoch 13/25
975/975 [==============================] - 160s 164ms/step - loss: 0.0146 - accuracy: 0.9962 - val_loss: 0.0229 - val_accuracy: 0.9950

Epoch 00013: val_loss did not improve from 0.01693
Epoch 14/25
975/975 [==============================] - 160s 164ms/step - loss: 0.0140 - accuracy: 0.9967 - val_loss: 0.1996 - val_accuracy: 0.9698

Epoch 00014: val_loss did not improve from 0.01693
Epoch 15/25
975/975 [==============================] - 159s 164ms/step - loss: 0.0327 - accuracy: 0.9941 - val_loss: 0.0380 - val_accuracy: 0.9911

Epoch 00015: val_loss did not improve from 0.01693
```

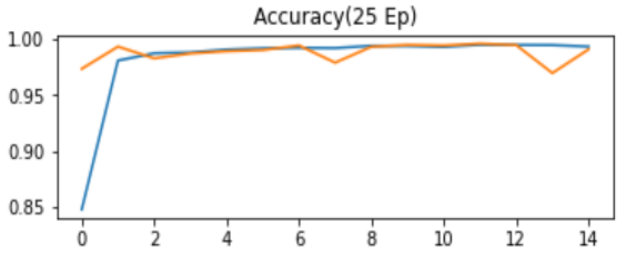
```
In [26]: model25.save(r"D:\jupyternotebooks\vgg16\model25weights")

         INFO:tensorflow:Assets written to: D:\jupyternotebooks\vgg16\model25weights\assets
```

```
In [28]: model_file="VGG16(25ep).h5"
         keras.models.save_model(model25,model_file)
```

In [29]:
```
plt.subplot(211)
plt.title('Loss(25 Ep)')
plt.plot(history25.history['loss'], label='train')
plt.plot(history25.history['val_loss'], label='test')
plt.legend()
plt.savefig('Loss(25 Ep).png')
plt.show()
```



In [30]:
```
plt.subplot(212)
plt.title('Accuracy(25 Ep)')
plt.plot(history25.history['accuracy'], label='train')
plt.plot(history25.history['val_accuracy'], label='test')
plt.savefig('Accuracy(25 Ep).png')
plt.show()
```



In [3]:
```
model25 = keras.models.load_model(r'D:\jupyternotebooks\vgg16\model25weights')
test=ImageDataGenerator(rotation_range=20,zoom_range=0.1,
                        width_shift_range=0.1,height_shift_range=0.1,
                        shear_range=0.1,horizontal_flip=True,
                        rescale=1/255.0).flow_from_directory(test_dir,target_size=(64,64),class_mode=None)
pred=model25.predict(test)
pred = np.argmax(pred, axis=1)
pred
```

```
Found 1 images belonging to 1 classes.
```

Out[3]:
```
array([3], dtype=int64)
```

In [3]:
```
converter = tf.lite.TFLiteConverter.from_saved_model(r'D:\jupyternotebooks\vgg16\model25weights')
tflite_model = converter.convert()

with open('model.tflite', 'wb') as f:
  f.write(tflite_model)
```

In [3]:
```
converter = tf.lite.TFLiteConverter.from_saved_model(r'D:\jupyternotebooks\vgg16\model25weights')
converter.optimizations = [tf.lite.Optimize.DEFAULT]
converter.representative_dataset = representative_dataset
tflite_model = converter.convert()

with open('model_quant.tflite', 'wb') as f:
  f.write(tflite_model)
```

In [4]:
```
img_gen=ImageDataGenerator(rotation_range=20,zoom_range=0.1,
                           width_shift_range=0.1,height_shift_range=0.1,
                           shear_range=0.1,horizontal_flip=True,
                           rescale=1/255.0,validation_split=0.2)
test_gen=img_gen.flow_from_directory(test_dir,target_size=(64,64),class_mode=None, batch_size=1, shuffle=False)

def represent_data_gen():
    for ind in range(len(test_gen.filenames)):
        img_with_label = test_gen.next() # it returns (image and label) tuple
        yield [np.array(img_with_label[0], dtype=np.float32, ndmin=2)] # return only image
```

```
Found 40 images belonging to 1 classes.
```

In [3]:
```
converter = tf.lite.TFLiteConverter.from_saved_model(r'D:\jupyternotebooks\vgg16\model25weights')
converter.optimizations = [tf.lite.Optimize.DEFAULT]

converter.representative_dataset = represent_data_gen
tflite_model = converter.convert()

tflite_both_quant_file =  "vgg16_quant.tflite"
with open(tflite_both_quant_file, "wb") as f:
    f.write(tflite_model)
```

7.4 Android Code

7.4.1 MainActivity.java

```
package com.example.sli;


import androidx.appcompat.app.AppCompatActivity;


import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;


public class MainActivity extends AppCompatActivity {
    private Button predictbtn;
    private Button speakbtn;
    private Button learnbtn;
    private Button aboutusbtn;
```

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    predictbtn=(Button)findViewById(R.id.predictbtn);
    predictbtn.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            openpredictactiv();
        }
    });
    speakbtn=(Button)findViewById(R.id.speakbtn);
    speakbtn.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            openspeakactiv();
        }
    });
    learnbtn=findViewById(R.id.learnbtn);
    learnbtn.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            openlearnactiv();
        }
    });
    aboutusbtn=findViewById(R.id.aboutusbtn);
    aboutusbtn.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            opensboutusactiv();
        }
    });
}
```

```
    public void openpredictactiv(){
        Intent intent =new Intent(this,predictactivity.class);
        startActivity(intent);
    }
    public void openspeakactiv(){
        Intent intent =new Intent(this,speakactivity.class);
        startActivity(intent);
    }
    public void openlearnactiv(){
        Intent intent=new Intent(this,learnactivity.class);
        startActivity(intent);
    }
    public void opensboutusactiv(){
        Intent intent=new Intent(this,aboutusactivitiy.class);
        startActivity(intent);
    }
}
```

7.4.2 MainActivity.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <Button
        android:id="@+id/predictbtn"
        android:layout_width="415dp"
        android:layout_height="185dp"
        android:text="Predict"
        android:textColor="#050505"
```

```
    android:textSize="50sp"
    app:backgroundTint="#FBFBFB"
    app:layout_constraintBottom_toTopOf="@+id/speakbtn"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.0" />

<Button
    android:id="@+id/speakbtn"
    android:layout_width="415dp"
    android:layout_height="185dp"
    android:text="Speak"
    android:textSize="50sp"
    app:backgroundTint="#000000"
    android:textColor="#FBFBFB"
    app:layout_constraintBottom_toTopOf="@+id/learnbtn"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintStart_toStartOf="parent" />

<Button
    android:id="@+id/learnbtn"
    android:layout_width="415dp"
    android:layout_height="185dp"
    android:text="Learn"
    android:textColor="#050505"
    android:textSize="50sp"
    app:backgroundTint="#FBFBFB"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintStart_toStartOf="parent"
```

```
        app:layout_constraintTop_toBottomOf="@+id/speakbtn" />

    <Button
        android:id="@+id/aboutusbtn"
        android:layout_width="415dp"
        android:layout_height="185dp"
        android:text="About us"
        android:textSize="50sp"
        app:backgroundTint="#000000"
        android:textColor="#FBFBFB"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/learnbtn"
        app:layout_constraintVertical_bias="0.0" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

7.4.3 Predict.java

```
package com.example.sli;
import com.example.sli.ml.Model;

import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;

import android.Manifest;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.graphics.Bitmap;
import android.os.Bundle;
import android.provider.MediaStore;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;
```

```
import android.widget.TextView;

import java.io.BufferedReader;
import java.io.DataInputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.io.InputStreamReader;
import java.nio.ByteBuffer;
import java.util.ArrayList;
import java.util.Locale;

import org.tensorflow.lite.DataType;
import org.tensorflow.lite.support.image.TensorImage;
import org.tensorflow.lite.support.tensorbuffer.TensorBuffer;
import android.speech.tts.TextToSpeech;

public class predictactivity extends AppCompatActivity {
   Button camerabtn;
   ImageView imageView;
   TextView textView;
   TextToSpeech textToSpeech;

   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_predictactivity);
      camerabtn=(Button)findViewById(R.id.camerabtn);
      imageView=(ImageView)findViewById(R.id.imageView);
      textView=findViewById(R.id.txtview);
      if (ContextCompat.checkSelfPermission(predictactivity.this,
Manifest.permission.CAMERA)!= PackageManager.PERMISSION_GRANTED){
```

```
            ActivityCompat.requestPermissions(predictactivity.this,new String[]{
                    Manifest.permission.CAMERA},
                    100);


        }
        camerabtn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent intent= new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
                startActivityForResult(intent,100);
            }
        });

    // Text to Speech
        textToSpeech = new TextToSpeech(getApplicationContext(), new
    TextToSpeech.OnInitListener() {
            @Override
            public void onInit(int i) {
                if(i!=TextToSpeech.ERROR){
                    textToSpeech.setLanguage(Locale.UK);
                }
            }
        });
    }
    public static int argmax(float[] args) {
        int result;
        result = 0;
        for (int i = 1; i < args.length; i++) {
            if (Float.compare(args[result], args[i]) < 0)
                result = i;
        }
        return result;
    }
```

```
public String readlabelfile(int pos){
    String label;
    File labelfile=new File("assets/","labels.txt");
    ArrayList<String> labellist=new ArrayList<String>();
    try{
        FileInputStream fin=new FileInputStream(labelfile);
        DataInputStream din=new DataInputStream(fin);
        BufferedReader bin=new BufferedReader(new InputStreamReader(din));
        String strline;
        while ((strline=bin.readLine())!=null){
            ArrayList<String> buff=new ArrayList<String>();
            buff.add(strline);
            labellist.addAll(buff);
        }
    } catch (FileNotFoundException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    }
    return labellist.get(pos);
}

public String labelcond(int pos){

@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    if (requestCode == 100) {
        Bundle bundle=data.getExtras();
        Bitmap captureimage = (Bitmap) data.getExtras().get("data");
        imageView.setImageBitmap(captureimage);
        Bitmap img =Bitmap.createScaledBitmap(captureimage,64,64,true);
```

```
            try {
                Model model = Model.newInstance(getApplicationContext());

                // Creates inputs for reference.
                TensorBuffer inputFeature0 = TensorBuffer.createFixedSize(new
int[]{1, 64, 64, 3}, DataType.FLOAT32);
                TensorImage tensorImage= new TensorImage(DataType.FLOAT32);
                tensorImage.load(img);
                ByteBuffer byteBuffer=tensorImage.getBuffer();
                inputFeature0.loadBuffer(byteBuffer);

                // Runs model inference and gets result.
                Model.Outputs outputs = model.process(inputFeature0);
                TensorBuffer outputFeature0 =
outputs.getOutputFeature0AsTensorBuffer();
                int labelpos=argmax(outputFeature0.getFloatArray());
                String label=labelcond(labelpos);
                textView.setText(label);
                textToSpeech.speak(label,TextToSpeech.QUEUE_FLUSH,null);

                // Releases model resources if no longer used.
                model.close();
            } catch (IOException e) {
                // TODO Handle the exception
            }
        }
    }
}
```

7.4.4 Predict.xml

```
package com.example.sli;

import androidx.annotation.RequiresApi;
import androidx.appcompat.app.AppCompatActivity;
```

```
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;

import android.Manifest;
import android.annotation.SuppressLint;

import android.content.Intent;
import android.content.pm.PackageManager;
import android.os.Build;
import android.os.Bundle;
import android.os.StrictMode;
import android.speech.RecognitionListener;
import android.speech.RecognizerIntent;
import android.speech.SpeechRecognizer;
import android.view.KeyEvent;
import android.view.View;
import android.webkit.WebView;
import android.webkit.WebViewClient;
import android.widget.EditText;
import android.widget.ImageView;

import com.android.volley.Request;
import com.android.volley.Response;
import com.android.volley.toolbox.StringRequest;
import com.android.volley.toolbox.Volley;

import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;

import java.io.IOException;
import java.util.ArrayList;
import okhttp3.MediaType;
```

```
import okhttp3.OkHttpClient;
import okhttp3.RequestBody;

public class speakactivity extends AppCompatActivity {
    WebView webView;
    EditText speechtext;
    ArrayList<String> buff_data=new ArrayList<>();
    OkHttpClient client = new OkHttpClient();
    @SuppressLint("SetJavaScriptEnabled")
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_speakactivity);
        String giphy_url="file:///android_asset/index.html";
        ImageView mic=findViewById(R.id.micimgview);
        speechtext=findViewById(R.id.speechedittext);
        SpeechRecognizer speechRecognizer;
        webView=findViewById(R.id.webview);
        webView.getSettings().setJavaScriptEnabled(true);
        webView.loadUrl(giphy_url);
        webView.setWebViewClient(new WebViewClient());
        buff_data.add("Hello");
        load_gif(buff_data);
        speechtext.setOnKeyListener(new View.OnKeyListener() {
            @Override
            public boolean onKey(View v, int keyCode, KeyEvent event) {
                if(event.getAction()==KeyEvent.ACTION_DOWN &&
keyCode==KeyEvent.KEYCODE_ENTER){
                    buff_data.remove(0);
                    buff_data.add(speechtext.getText().toString());
                    load_gif(buff_data);
                    return true;
                }
```

```
                return false;
            }
        });
        if(ContextCompat.checkSelfPermission(this,
Manifest.permission.RECORD_AUDIO)!=
PackageManager.PERMISSION_GRANTED){
            ActivityCompat.requestPermissions(this,new String[]{
                Manifest.permission.RECORD_AUDIO
            },1);
        }
        speechRecognizer=SpeechRecognizer.createSpeechRecognizer(this);
        Intent speechrecognizerintent=new
Intent(RecognizerIntent.ACTION_RECOGNIZE_SPEECH);
        final int[] count = {0};
        mic.setOnClickListener(new View.OnClickListener() {
            @RequiresApi(api = Build.VERSION_CODES.LOLLIPOP)
            @Override
            public void onClick(View v) {
                if(count[0] ==0){
                    mic.setImageDrawable(getDrawable(R.drawable.redmic));
                    speechRecognizer.startListening(speechrecognizerintent);
                    count[0] =1;
                }
                else{
                    mic.setImageDrawable(getDrawable(R.drawable.micon));
                    speechRecognizer.stopListening();
                    count[0] =0;
                }
            }
        });
        speechRecognizer.setRecognitionListener(new RecognitionListener() {
            @Override
            public void onReadyForSpeech(Bundle params) {
```

```
            }
            @Override
            public void onBeginningOfSpeech() {
            }
            @Override
            public void onRmsChanged(float rmsdB) {
            }
            @Override
            public void onBufferReceived(byte[] buffer) {
            }
            @Override
            public void onEndOfSpeech() {
            }
            @Override
            public void onError(int error) {
            }
            @Override
            public void onResults(Bundle results) {
               ArrayList<String> data
=results.getStringArrayList(speechRecognizer.RESULTS_RECOGNITION);
               speechtext.setText(data.get(0));
               load_gif(data);
            }
            @Override
            public void onPartialResults(Bundle partialResults) {
            }
            @Override
            public void onEvent(int eventType, Bundle params) {
            }
         });
      }
      @Override
```

```
    public void onRequestPermissionsResult(int requestCode, String[] permissions,
int[] grantResults) {
        super.onRequestPermissionsResult(requestCode, permissions, grantResults);
        if (requestCode == 1) {
            if (grantResults[0] == PackageManager.PERMISSION_GRANTED) {
//                Toast.makeText(this,"Permission Granted",Toast.LENGTH_SHORT);
            }
            else{
//                Toast.makeText(this,"Permission Denied",Toast.LENGTH_SHORT);
            }
        }
    }
    public void load_gif(ArrayList<String> data){
        String url="file:///android_asset/index.html";
        String searchterm= data.get(0);
        speechtext.setText(searchterm);
        webView.getSettings().setJavaScriptEnabled(true);
        webView.loadUrl(url);
        webView.setWebViewClient(new WebViewClient(){
            public void onPageFinished(WebView view, String url){
                webView.loadUrl("javascript:spellcheckngify('" + searchterm + "')");
            }
        });
    }
}
```

7.4.5 Speak.java

```
package com.example.sli;

import androidx.annotation.RequiresApi;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;
```

```
import android. Manifest;
import android.annotation.SuppressLint;

import android.content.Intent;
import android.content.pm.PackageManager;
import android.os.Build;
import android.os.Bundle;
import android.os.StrictMode;
import android.speech.RecognitionListener;
import android.speech.RecognizerIntent;
import android.speech.SpeechRecognizer;
import android.view.KeyEvent;
import android.view.View;
import android.webkit.WebView;
import android.webkit.WebViewClient;
import android.widget.EditText;
import android.widget.ImageView;

import com.android.volley.Request;
import com.android.volley.Response;
import com.android.volley.toolbox.StringRequest;
import com.android.volley.toolbox.Volley;

import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;

import java.io.IOException;
import java.util.ArrayList;

import okhttp3.MediaType;
import okhttp3.OkHttpClient;
import okhttp3.RequestBody;
```

```
public class speakactivity extends AppCompatActivity {
    WebView webView;
    EditText speechtext;
    ArrayList<String> buff_data=new ArrayList<>();
    OkHttpClient client = new OkHttpClient();
    @SuppressLint("SetJavaScriptEnabled")
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_speakactivity);
        String giphy_url="file:///android_asset/index.html";
        ImageView mic=findViewById(R.id.micimgview);
        speechtext=findViewById(R.id.speechedittext);
        SpeechRecognizer speechRecognizer;
        webView=findViewById(R.id.webview);
        webView.getSettings().setJavaScriptEnabled(true);
        webView.loadUrl(giphy_url);
        webView.setWebViewClient(new WebViewClient());
        buff_data.add("Hello");
        load_gif(buff_data);
        speechtext.setOnKeyListener(new View.OnKeyListener() {
            @Override
            public boolean onKey(View v, int keyCode, KeyEvent event) {
                if(event.getAction()==KeyEvent.ACTION_DOWN &&
keyCode==KeyEvent.KEYCODE_ENTER){
                    buff_data.remove(0);
                    buff_data.add(speechtext.getText().toString());
                    load_gif(buff_data);
                    return true;
                }
                return false;
            }
```

```
        });
        if(ContextCompat.checkSelfPermission(this,
    Manifest.permission.RECORD_AUDIO)!=
    PackageManager.PERMISSION_GRANTED){
            ActivityCompat.requestPermissions(this,new String[]{
                Manifest.permission.RECORD_AUDIO
            },1);
        }
        speechRecognizer=SpeechRecognizer.createSpeechRecognizer(this);
        Intent speechrecognizerintent=new
    Intent(RecognizerIntent.ACTION_RECOGNIZE_SPEECH);
        final int[] count = {0};
        mic.setOnClickListener(new View.OnClickListener() {
            @RequiresApi(api = Build.VERSION_CODES.LOLLIPOP)
            @Override
            public void onClick(View v) {
                if(count[0] ==0){
                    mic.setImageDrawable(getDrawable(R.drawable.redmic));
                    speechRecognizer.startListening(speechrecognizerintent);
                    count[0] =1;
                }
                else{
                    mic.setImageDrawable(getDrawable(R.drawable.micon));
                    speechRecognizer.stopListening();
                    count[0] =0;
                }
            }
        });
        speechRecognizer.setRecognitionListener(new RecognitionListener() {
            @Override
            public void onReadyForSpeech(Bundle params) {
            }
```

```
            @Override
            public void onBeginningOfSpeech() {
            }
            @Override
            public void onRmsChanged(float rmsdB) {
            }
            @Override
            public void onBufferReceived(byte[] buffer) {
            }
            @Override
            public void onEndOfSpeech() {
            }
            @Override
            public void onError(int error) {
            }
            @Override
            public void onResults(Bundle results) {
               ArrayList<String> data
=results.getStringArrayList(speechRecognizer.RESULTS_RECOGNITION);
               speechtext.setText(data.get(0));
               load_gif(data);
            }
            @Override
            public void onPartialResults(Bundle partialResults) {
            }
            @Override
            public void onEvent(int eventType, Bundle params) {
            }
         });
      }
      @Override
      public void onRequestPermissionsResult(int requestCode, String[] permissions,
int[] grantResults) {
```

```
        super.onRequestPermissionsResult(requestCode, permissions, grantResults);
        if (requestCode == 1) {
            if (grantResults[0] == PackageManager.PERMISSION_GRANTED) {
//              Toast.makeText(this,"Permission Granted",Toast.LENGTH_SHORT);
            }
            else{
//              Toast.makeText(this,"Permission Denied",Toast.LENGTH_SHORT);
            }
        }
    }
    public void load_gif(ArrayList<String> data){
        String url="file:///android_asset/index.html";
        String searchterm= data.get(0);
        speechtext.setText(searchterm);
        webView.getSettings().setJavaScriptEnabled(true);
        webView.loadUrl(url);
        webView.setWebViewClient(new WebViewClient(){
            public void onPageFinished(WebView view, String url){
                webView.loadUrl("javascript:spellcheckngify('" + searchterm + "')");
            }
        });
    }
}
```

7.4.6 Speak.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".speakactivity">
```

```
<WebView
    android:id="@+id/webview"
    android:layout_width="418dp"
    android:layout_height="471dp"
    android:layout_marginTop="4dp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

<EditText
    android:id="@+id/speechedittext"
    android:layout_width="240dp"
    android:layout_height="129dp"
    android:layout_marginTop="8dp"
    android:layout_marginEnd="8dp"
    android:hint="Enter the word!"
    android:imeOptions="actionSearch"
    android:singleLine="true"
    android:textColor="#050505"
    android:textSize="25sp"
    android:textStyle="bold"
    app:layout_constraintEnd_toStartOf="@+id/micimgview"
    app:layout_constraintTop_toBottomOf="@+id/webview" />
<ImageView
    android:id="@+id/micimgview"
    android:layout_width="105dp"
    android:layout_height="172dp"
    android:layout_marginTop="8dp"
    android:layout_marginEnd="4dp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/webview"
    app:srcCompat="@drawable/micon" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

TESTING

Chapter 8

TESTING

Unit testing is a development procedure where programmers create tests as they develop software. The tests are simple short tests that test functionally of a particular unit or module of their code, such as a class or function.Using open source libraries like cunit, oppunit and nun it (for C, C++ and C#) these tests can be automatically run and any problems found quickly. As the tests are developed in parallel with the source unit test demonstrates its correctness.

8.1 Validation and System Testing

Validation testing is a concern which overlaps with integration testing. Ensuring that the application fulfils its specification is a major criterion for the construction of an integration test. Validation testing also overlaps to a large extent with System Testing, where the application is tested with respect to its typical working environment. Consequently, for many processes no clear division between validation and system testing can be made. Specific tests which can be performed in either or both stages include the following.

- **Regression Testing**: Where this version of the software is tested with the automated test harness used with previous versions to ensure that the required features of the previous version are skill working in the new version.
- **Recovery Testing**: Where the software is deliberately interrupted in a number of ways off, to ensure that the appropriate techniques for restoring any lost data will function.
- **Security Testing**: Where unauthorized attempts to operate the software, or parts of it, attempted it might also include attempts to obtain access the data, or harm the software installation or even the system software. As with all types of security determined will be able to obtain unauthorized access and the best that can be achieved is to make this process as difficult as possible.
- **Stress Testing** : where abnormal demands are made upon the software by increasing the rate at which it is asked to accept, or the rate at which it is asked to produce information. More complex tests may attempt to create very large data sets or cause the softwares to make excessive demands on the operating system
- **Performance Testing**: Where the performance requirements, if any, are checked. These may include the size of the software when installed, type amount of main memory and/or secondary storage it requires and the demands made of the operating when running with normal limits or the response time.

- **Usability Testing**: The process of usability measurement was introduced in the previous chapter. Even if usability prototypes have been tested whilst the application was constructed, a validation test of the finished product will always be required.
- **Alpha and Beta Testing**: This is where the software is released to the actual end users. An initial release, the alpha release, might be made to selected users who be expected to report bugs and other detailed observations back to the production team. Once the application changes necessitated by the alpha phase can be made to larger more representative set users, before the final release is made to all users.

The final process should be a Software audit where the complete software project is checked to ensure that it meets production management requirements. This ensures that all required documentation has been produced, is in the correct format and is of acceptable quality. The purpose of this review is: firstly to assure the quality of the production process and by implication construction phase commences. A formal hand over from the development team at the end of the audit will mark the transition between the two phases.

- **Integration Testing**: Integration Testing can proceed in a number of different ways, which can be broadly characterized as top down or bottom up. In top down integration testing the high level control routines are tested first, possibly with the middle level control structures present only as stubs. Subprogram stubs were presented in section2 as incomplete subprograms which are only present to allow the higher. Level control routines to be tested. Top down testing can proceed in a depth-first or a breadth-first manner. For depth- first integration each module is tested in increasing detail, replacing more and more levels of detail with actual code rather than stubs. Alternatively, breadth-first would processed by refining all the modules at the same level of control throughout the application .in practice a combination of the two techniques would be used.

  At the initial stages all the modules might be only partly functional, possibly being implemented only to deal with non- erroneous data. These would be tested in breadth-first manner, but over a period of time each would be replaced with successive refinements which were closer to the full functionality. This allows depth-first testing of a module to be performed simultaneously with breadth- first testing of all the modules.

- **Unit Testing**: Unit testing deals with testing a unit as a whole. This would test the interaction of many functions but confine the test within one unit. The exact scope of a unit is left to interpretation. Supporting test code, sometimes called Scaffolding, may be necessary to support an individual test. This type of testing is driven by the architecture and implementation teams. This focus is also called black-box testing because only the details of the interface are visible to the test. Limits that are global to a unit are tested here.

  In the construction industry, scaffolding is a temporary, easy to assemble and disassemble, frame placed around a building to facilitate the construction of the building. The construction workers first build the scaffolding and then the building. Later the scaffolding is removed, exposing the completed building. Similarly, in software testing, one particular test may need some supporting software. This software establishes can a correct evaluation of the test take place. The scaffolding software may establish state and values for data structures as well as providing dummy external functions for the test. Different scaffolding software may be needed form one test to another test. Scaffolding software rarely is considered part of the system.

  Sometimes the scaffolding software becomes larger than the system software being tested. Usually the scaffolding software is not of the same quality as the system software and frequently is quite fragile. A small change in test may lead to much larger changes in the scaffolding. Internal and unit testing can be automated with the help of coverage tools. Analyzes the source code and generated a test that will execute every alternative thread of execution. Typically, the coverage tool is used in a slightly different way. First the coverage tool is used to augment the source by placing information prints after each line of code. Then the testing suite is executed generating an audit trail. This audit trail is analyzed and reports the percent of the total system code executed during the test suite. If the coverage is high and the untested source lines are of low impact to the system‘s overall quality, then no more additional tests are required.

8.1.1 Test case for Main Page

SL No	Actions	Inputs	Excpeted output	Actual output	Test result
1	Open the activity which is associated with the button	Press main buttons	Open the new activity	Acitivity is opened	Pass

Table 8.1: Test case for Main Page

8.1.2 Test case for Predict Page

SL No	Actions	Inputs	Excpeted output	Actual output	Test result
1	Open the camera interface, captures an images and display it	Press the camera button	Camera opens and captures the image	Camera interface works and displays the image	Pass
2	Model Prediction	Image is passed to the function	Model converts the images into bitmap and predicts the output	Models predicts the output and displays the output with bitmap image	Pass
3	Text to Speech	The model predication output	Convert the output to speech	Converts the output to speech effectively	Pass

Table 8.2: Test case for Predict Page

8.1.3 Test case for Speak Page

SL No	Actions	Inputs	Excpeted output	Actual output	Test result
1	Spell check model	Text from the user	Model corrects the words and passes the giphy	Model does it work and passed to giphy function	Pass
2	Model correction message	From the spell check model	Message is displayed	Message is displayed as the suggestions	Pass
3	Gif is retrieved	Passed from the spell correction model	Gifs are retrieved and displayed	Gifs are retrieved and displayed	Pass

4	Speech to text	Recorded from the user	The recorded audio is converted into text is sent to the giphy function	The audio is converted into text and sent to the giphy function	Pass

Table 8.3: Test case for Speak Page

8.2 Testing Summary

This chapter shows the various test results produced by the system. Various kinds of test are performed for each part of the system and as well as the whole system. It shows various pre-defined test cases and result of running these test cases on the system. It provides the comparison of expected output and the actual output produced by system based on which bugs are identified and eliminated.

SCREENSHOTS

Chapter 9

SCREENSHOTS

9.1 Main Page

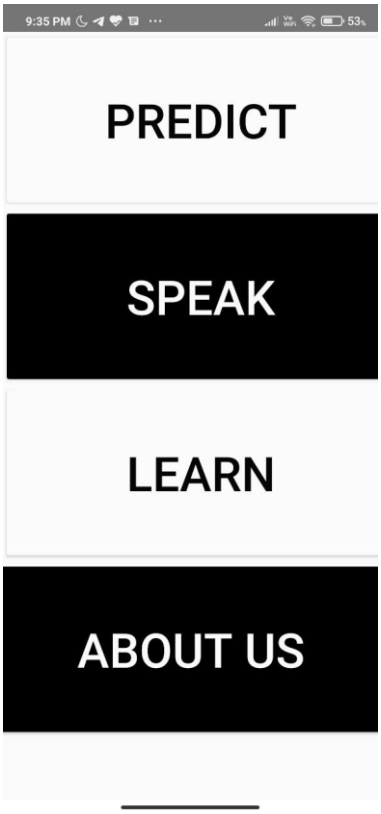


Figure 9.1: Main Page

9.2 Camera Interface

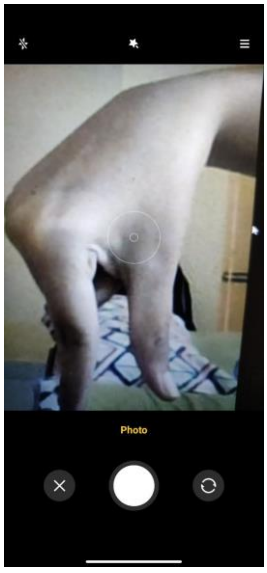


Figure 9.2: Camera Interface

9.3 Predict Page

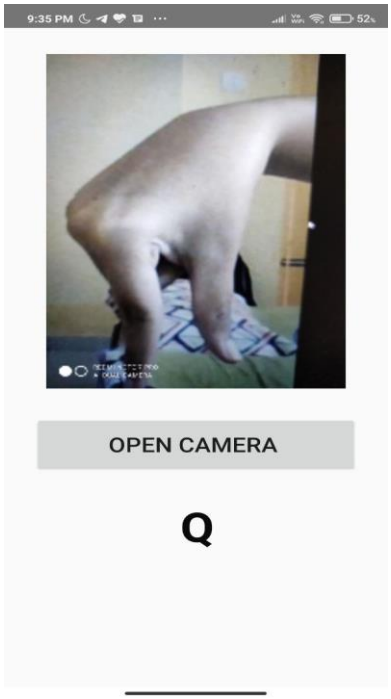


Figure 9.3: Predict Page

9.4 Speak page with input from edit text method

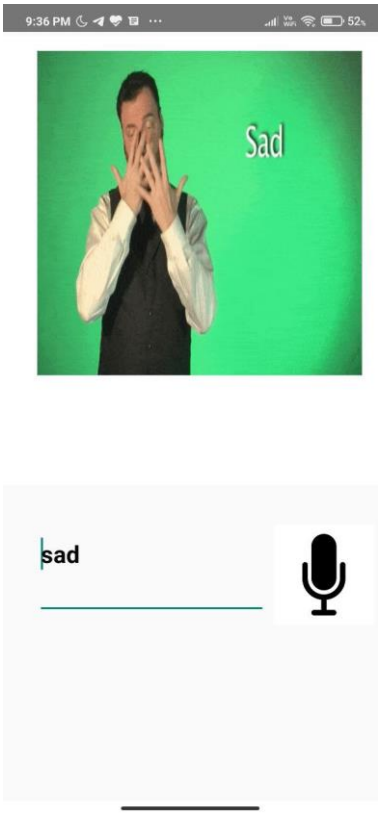


Figure 9.4: Speak page with input from edit text method

9.5 Speak page with audio method input

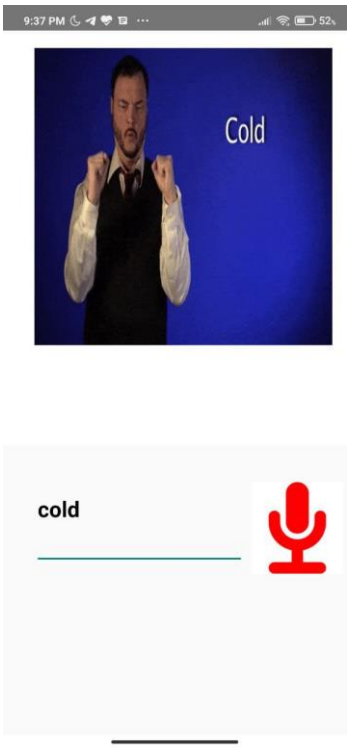


Figure 9.5: Speak page with audio method input

9.6 Speak page with spell check model suggestions

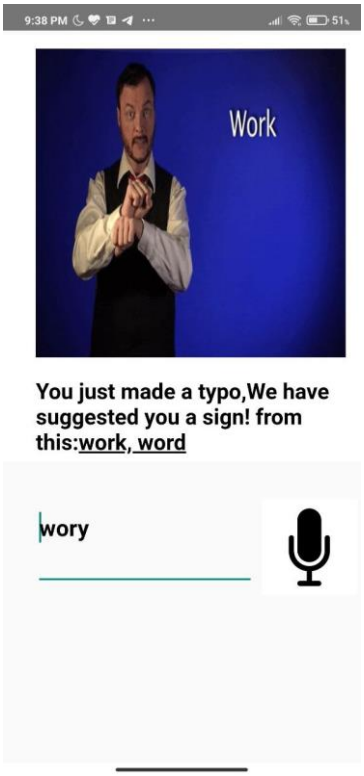


Figure 9.6: Speak page with spell check model suggestions

9.7 Learn Page

Figure 9.7: Learn Page

9.8 Learn Page Connector

Figure 9.8: Learn Page Connector

9.9 About Us Page

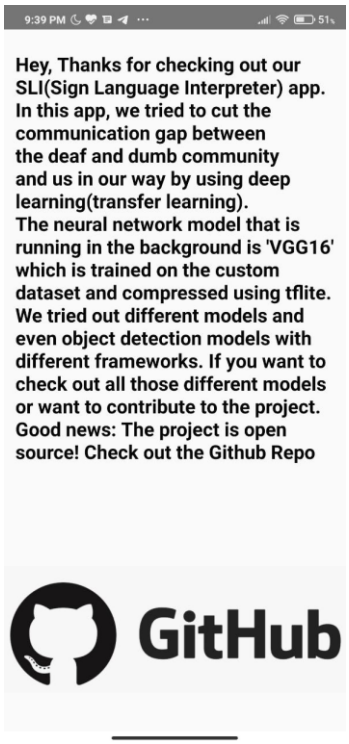


Figure 9.9: About Us Page

9.1 Sample Output Summary

This chapter shows various screenshots of the system. It also shows how data processing happens at various stages of the system and the final output is also displayed. And it also shows the outer interface design of the system.

CONCLUSION

CONCLUSION

A sign language is a natural mode of communication used by the deaf community. India has large population of speech and hearing impaired but a very small number of certified sign language interpreters are available. Research in hand gesture recognition has gained attention with advancement in the field of computer vision. A Sign Language Interpreter (SLI) App decodes and understands the information conveyed by signs. SLI App can be a major breakthrough in helping a common people to communicate with the deaf and can help in bridging this communication gap. A SLI App can be designed based on video/image processing and deep learning techniques which requires a standard dataset, determination of an optimal feature set, and an appropriate classification technique.

BIBLIOGRAPHY

REFERENCES

Reference Papers:

[1] Authors: Yaofeng Xue; Shang Gao; Huali Sun; Wei Qin

"A Chinese Sign Language Recognition System Using Leap Motion"

Published in: 2017 International Conference on Virtual Reality and Visualization (ICVRV)

Link: https://ieeexplore.ieee.org/document/8719155

[2] Authors: Xue Wu; Xiao-ru Song; Song Gao; Chao-bo Chen

"Gesture recognition based on transfer learning"

Published in: 2019 Chinese Automation Congress (CAC)

Link: https://ieeexplore.ieee.org/document/8997255

[3] Authors: Sarfaraz Masood,Harish Chandra Thuwal,Adhyan Srivastava

"American Sign Language Character Recognition Using Convolutional Neural Network"

Published in: In book: Smart Computing and Informatics (pp.403-412)

Link:

https://www.researchgate.net/publication/320703517_American_Sign_Language_Character_Recognition_Using_Convolution_Neural_Network

[4] Authors: Hongyang Gao; Zhengyang Wang; Lei Cai; Shuiwang Ji

"ChannelNets: Compact and Efficient Convolutional Neural Networks via Channel-Wise Convolutions"

Published in: IEEE Transactions on Pattern Analysis and Machine Intelligence (Early Access)

Link: https://ieeexplore.ieee.org/document/9007485

[5] Authors: Qingqing Cao, Niranjan Balasubramanian, Aruna Balasubramanian

"MobiRNN: Efficient Recurrent Neural Network Execution on Mobile GPU"

Published at 1st International Workshop on Embedded and Mobile Deep Learning colocated with MobiSys 2017

Link: https://arxiv.org/abs/1706.00878

[6] Authors: Chirag Patel,Ripal Patel

"Gaussian mixture model based moving object detection from video sequence"

Conference: Proceedings of the ICWET '11 International Conference & Workshop on Emerging Trends in Technology

Link:
https://www.researchgate.net/publication/220902168_Gaussian_mixture_model_based_moving_object_detection_from_video_sequence

[7] Authors: Huy B.D Nguyen; Hung Ngoc Do

"Deep Learning for American Sign Language Fingerspelling Recognition System"

Published in: 2019 26th International Conference on Telecommunications (ICT)

Link: https://ieeexplore.ieee.org/document/8798856

[8] Auhtors: Wang Nan; Zhou Zhigang; Lei Huan; Ma Jingqi; Zhuang Jiajun; Duan Guangxue

"Gesture Recognition Based on Deep Learning in Complex Scenes"

Published in: 2019 Chinese Control And Decision Conference (CCDC)

Link: https://ieeexplore.ieee.org/document/8833349

[9] Authors: Dhruv Rathi

"Optimization of Transfer Learning for Sign Language Recognition Targeting Mobile Platform"

Published as Computer Vision and Pattern Recognition (cs.CV)

Link: https://arxiv.org/abs/1805.06618

[10] Authors: Niels Martínez-Guevara; José-Rafael Rojano-Cáceres; Arturo Curiel

"Detection of Phonetic Units of the Mexican Sign Language"

Published in: 2019 International Conference on Inclusive Technologies and Education (CONTIE)

Link: https://ieeexplore.ieee.org/document/8971399

[11] Authors: Jungheon Lee;Youngsan Koh;Jihoon Yang

"A Deep Learning Based Video Classification System
Using Multimodality Correlation Approach"

Published in: 2017 17th International Conference on Control, Automation and Systems (ICCAS)

Link: https://ieeexplore.ieee.org/document/8204286

[12] Authors: Sanjay Kumar;Manish Kumar

"A Study on the Image Detection Using Convolution Neural Networks and TenserFlow"

Published in: 2018 International Conference on Inventive Research in Computing Applications (ICIRCA)

Link: https://ieeexplore.ieee.org/document/8597204

[13] Authors: Mathieu Virbel1,Thomas Hansen2,Oleksandr Lobunets3

"Kivy – A Framework for Rapid Creation of Innovative User Interfaces"

Published in:

Link:

https://dl.gi.de/bitstream/handle/20.500.12116/8013/Virbel_Hansen_Lobunets_2011.pdf?sequence=2&isAllowed=y

[14] Authors: Ahmed Fawzy Gad

"NumPyCNNAndroid: A Library for Straightforward Implementation of Convolutional Neural Networks for Android Devices"

Published in: 2019 International Conference on Innovative Trends in Computer Engineering (ITCE)

Link: https://ieeexplore.ieee.org/document/8646653

[15] Authors: Soeb Hussain;Rupal Saxena;Xie Han;Jameel Ahmed Khan;Hyunchul Shin

"Hand Gesture Recognition Using Deep Learning"

Published in: 2017 International SoC Design Conference (ISOCC)

Link: https://ieeexplore.ieee.org/document/8368821

[16] Authors: Shivashankara S,Srinath S

"American Sign Language Video Hand Gestures Recognition using Deep Neural Networks"

Published in: International Journal of Engineering and Advanced Technology (IJEAT) ISSN:2249-8958,Volume-8 Issue-5,June 2019

Link: https://www.ijeat.org/wp-content/uploads/papers/v8i5/E7205068519.pdf

[17] Authors: Rahul Chauhan;Kamal Kumar Ghanshala;R.C Joshi

"Convolutional Neural Network (CNN) for Image Detection and Recognition"

Published in: 2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC)

Link: https://ieeexplore.ieee.org/document/8703316

[18] Authors: Kshitij Bantupalli;Ying Xie

"American Sign Language Recognition using Deep Learning and Computer Vision"

Published in: 2018 IEEE International Conference on Big Data (Big Data)

Link: https://ieeexplore.ieee.org/document/8622141

[19] Authors: Mateen Ahmed;Mujtaba Idrees;Zain ul Abideen;Rafia Mumtaz;Sana Khalique

"Deaf Talk Using 3D Animated Sign Language A Sign Language Interpreter using Microsoft's Kinect v2"

Published in:2016 SAI Computing Conference (SAI)

Link: https://ieeexplore.ieee.org/document/7556002

[20] Authors: Brandon Garcia,Sigberto Alarcon Viesca

"Real-time American Sign Language Recognition with Convolutional Neural Networks"

Published in:

Link: http://cs231n.stanford.edu/reports/2016/pdfs/214_Report.pdf

Reference Websites:

[1] https://www.tensorflow.org/

[2] https://kivy.org/#home

[3] https://www.deeplearning.ai/

[4] https://ieeexplore.ieee.org/Xplore/home.jsp

[5] https://github.com/kivy/buildozer

Reference Text books:

[1] "Machine Learning Yearning", by Andrew Ng

[2] "Machine Learning Techniques ", by Tom M Mitchell

[3] "AI and Machine Learning for Coders: A Programmer's Guide to Artificial Intelligence" by Laurence Moroney

[4] "Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems", by Aurelien Geron

BASE PAPER

Automated Hand Gesture Recognition using a Deep Convolutional Neural Network model

Ishika Dhall
Department of Computer Science & Engineering
Amity University, Uttar Pradesh
India
ishikadhall11@gmail.com

Shubham Vashisth
Department of Computer Science & Engineering
Amity University, Uttar Pradesh
India
shubham.vashisth.delhi@gmail.com

Garima Aggarwal
Department of Computer Science & Engineering
Amity University, Uttar Pradesh
India
gmehta@amity.edu

***Abstract*—The tremendous growth in the domain of deep learning has helped in achieving breakthroughs in computer vision applications especially after convolutional neural networks coming into the picture. The unique architecture of CNNs allows it to extract relevant information from the input images without any hand-tuning. Today, with such powerful models we have quite a flexibility build technology that may ameliorate human life. One such technique can be used for detecting and understanding various human gestures as it would make the human-machine communication effective. This could make the conventional input devices like touchscreens, mouse pad, and keyboards redundant. Also, it is considered as a highly secure tech compared to other devices. In this paper, hand gesture technology along with Convolutional Neural Networks has been discovered followed by the construction of a deep convolutional neural network to build a hand gesture recognition application.**

***Keywords*—Convolutional Neural Networks, Hand Gesture Recognition system, Feature Map, Deep neural network**

I. INTRODUCTION

A gesture is a body movement that conveys a noteworthy implication. Gesture recognition is a computer science technology that helps a user in interacting with their digital devices using simple and natural body gestures. Gesture recognition technology can be beneficial at many places like automated home appliances, hand signal interpretation [1], automobiles, etc. Hand gesture recognition is a part of gesture recognition that is based on recognizing the movements of hands meant to be delivered, for example: showing a forefinger could denote the number "1" or a thumbs up could be an indication of agreement.

Deep learning is a fragment of a wide-ranging family of Artificial Intelligence. It essentially puts a light on the concept of a multi-layer perceptron learning. A Convolutional Neural Network commonly known as a Comp Net is a neural network class used in deep learning which is most applied to images and videos for their analysis. A CNN is a technique, or a machine learning model that can be applied to images to make them interpretable by machines. It can be implemented in other data analysis and classification problems as well. It is a type of artificial neural network which has a specialty of being able to deduce or distinguish patterns and understanding them. It is different than other deep learning models as it has an extra set of hidden layers called the convolutional layers along with the standard hidden layers. It can have one or more than one convolutional layer followed by the fully connected layers. The system will be learning features from each gesture and then further classify it. The entire notion of making a machine learn and making it smart is based on the abundance of data or information.

Our data fuels the machines and is used to make the machine learn to make predictions. The main aim of this paper is to train an algorithm which enables it to classify images of various hand gestures and signs like thumbs up, bolted fist, finger count, etc. Since the analysis of visual imaginings is being used, the class used to perform deep learning will be Convolutional Neural Network with Keras and TensorFlow as it is the standardized version of a multilayer perceptron.

This research provides the reader with a profound knowledge of a deep convolutional neural network. Also, this paper uses the data captured using the OpenCV library which will contribute to improving the accuracy score of the existing hand gesture recognition techniques.

In this research, a real-time anti-encroaching hand gesture recognition and hand tracking mechanism has been proposed which will improve the human-computer interactions and bring ease for the ones who rely on gestures for their day-to- day communication. It can be a significant communication tool for deafened people and people with ASD or autism spectrum disorder. It can be of great assistance for SOS signaling.

II. LITERATURE REVIEW

A technique of hand gesture recognition on a video game-based application has been proposed in [1].

978-1-7281-2791-0/20/$31.00 ©2020 IEEE

A new algorithm has been discussed in [1] to recognize and track hand gestures for better interaction with a video game. It is consisting of four hand gestures and four-hand direction classes to fulfil requirements that could have been extended to make it more powerful. It uses segmentation and tracking [2]. The proposed algorithm was performed on 40 samples and the accuracy turned out to be quite impressive.

Use of a convolutional neural network to reduce the feature extraction process and parameters being used has been discussed in [2]. The hand gesture recognition is performed using a convolutional neural network but the one used in our paper shows a deep convolutional neural network implementation. Results shown in [3] are very impressive when a training set of 50% of the database is used. Max Pooling Convolutional Neural Network to advance Human- robot interactions using color segmentation, edge blurring with morphological digital image processing and then experimenting with mobile robots using ARM 11 533 MHz [3].

They manage to get an accuracy score of around 96%. The vocabulary in the proposed project was up to 11 classes. It could have been used in a milieu of human-swarm interaction. Driver's hand gesture recognition via a 3D convolutional neural net was followed [4]. It engages spatial data augmentation techniques and pre-processing techniques for better results.

They achieved a score of 77% which could have been improved by constructing a deeper neural network. The challenges of a 3D CNN to perform classification and detection on the given dataset was addressed in [5] which also introduces a multi-modal dynamic challenging dataset and achieved an accuracy of 83.8%.

III. PRELIMINARIES

A convolutional neural network is a deep learning neural network class which is most applied to images and videos for its analysis. It is a kind of artificial neural network using machine learning algorithms for a unit and perceptron for supervised problems.

A CNN is basically a technique, or a machine learning model applied to images to make them interpretable by machines. It can have one or more than one convolutional layer followed by the fully connected layers. All types of cognitive tasks are performed using CNNs like Natural Language processing, image processing, etc. The concept of machine learning is not a contemporary thing, the first Artificial Intelligence-based program which came into play with a learned version of a game in which an Artificial Intelligence program was built that understood natural language finally.

In the given Fig 1, the image given as the input is used to find the primitive features like horizontal and vertical lines.

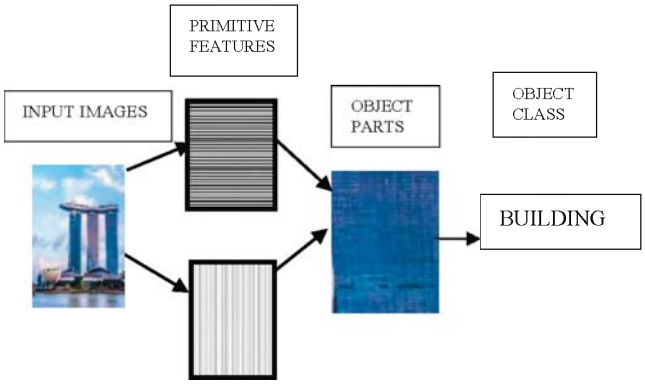


Fig 1. Working of a CNN Model

After the primitive feature extraction phase, the next stage determines the part of the given object using the features extracted. Objects parts are then used to interpret the class of the object.

Interfaces that cannot be touched not only improves the driver's focus and prevent possible mishaps but also makes the devices much more user-friendly due to which implementation of such technologies in several control systems is preferred.

IV. METHODOLOGY

Fig 2 shows a convolutional Neural Network model and various layers involved.

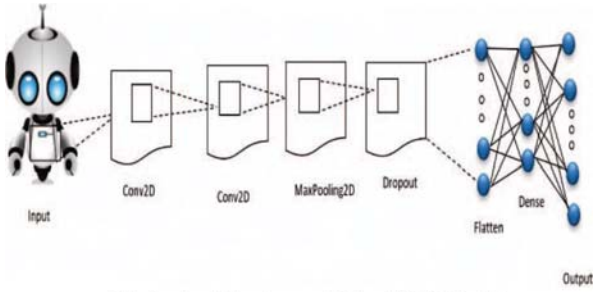


Fig 2. Architecture of the CNN Model

CNN is a type of neural network that is empowered with certain specific layers, such as:

1. Input layer
2. Hidden layer:

2.1. A Convolutional Layer
2.2. A Max Pooling Layer
2.3. A Fully Connected Layer
3. Output layer

A convolutional layer is the first Hidden Layer of convolutional deep learning and its key purpose is to detect and extract features from the image that was the input to our model like the edges and vertices of an image. Let's take an example, assume that we have an input image to be detected, then the hidden convolutional layer will help the system in finding the edges of that image. After that, we define a filter matrix also called a 'Kernel' which is further used for finding a new image with all the edges. Slide this filter for certain strides all over the image to find a new image with just the edges. Apply the dot product over the pixels of the image to find the image with all the edges. Edge image is useful for the initial steps and layers of CNN, in fact, it is the very first set of primitive feature sets for the working in a hierarchy of all the features.

The convolutional process is about detecting the edges, small patterns, orientations present in an image and it is present as a mathematical function. Convert the image into a matrix of binary integer values, i.e., 0s and 1s where appoint the value 0 to all the places where the image is black and appoint 1 to all the white parts of the image. Commonly while making this matrix, a value between 0 and 255 is used where all these numbers represent different shades of grey in case of a grayscale image. A three-channel image is used in case of input images which are not in grayscale format but are colored. Randomly select the type of filter and arbitrarily initialize the value of matrix elements.

The values of matrix elements are updated by optimum values as it goes through the training phase. Take dot product of pixels with every matrix value, now slide the kernel window over the image and find a new matrix called convolved image. Perform the element-wise multiplication of the kernel matrix and the image matrix. Try to apply multiple kernels in one image. Multiple kernels being applied to one image will result in multiple convolved matrices.

Leveraging different kernels assistances, it helps to find divergent patterns present in the image like curves, edges, etc. At the end of this layer, we receive a feature map which is the output of the process of convolution. Rectified linear unit (ReLu) [13] which is a non- linear activation function unit for providing a mapping between the response variables and inputs. Swap all negative numbers with a '0' and all positive numbers stay the same while using ReLu function (tf.nn.relu(x)).

For a pooling layer, down sample the output images for relu function to perform dimensionality reduction for the activated neurons. Use the Max Pooling layer to perform dimensionality reduction. Max Pooling finds the extreme values in input and simplifies the inputs. It diminishes the number of parameters within the model and converts lower-level data into a piece of higher-level information. In Fig 3, process of extracting a feature map is shown.

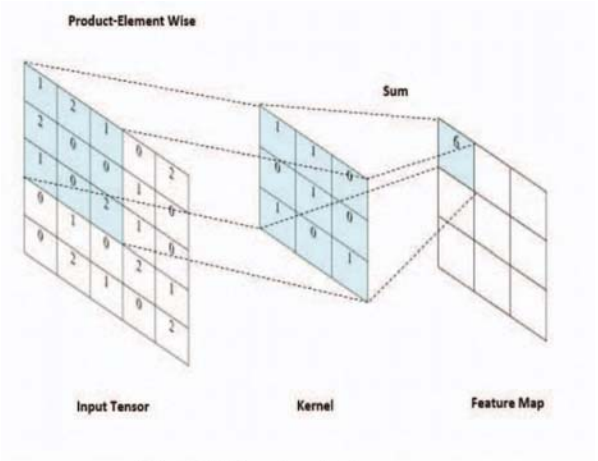


Fig 3. Extracting feature maps

For example, A 2x2 matrix results into a single pixel data by choosing the maximum value of the matrix. Strides dictates the sliding behavior of a max-pooling process [6]. It prevents overlapping, for example, if strides have a value two then the window will move 2 pixels every time. A fully connected layer takes the high-level images from the previous layer's filtered output and converts it into a vector. Each layer of the previous matrix will be first converted into a single (flatten) dimensional vector then each vector is fully connected to the next layer that is connected through a weight matrix [7]. SOFTMAX is an activation function that helps to find the class of each digit and generate probability for outputs.

The advantages of CNN include being a strong and computationally fast model, more efficient in terms of memory and it shows accurate results on images.

The disadvantages of CNN are that it is computationally expensive, quite complex, requires GPU and a large dataset along with the problem of Overfitting. It can also result in an imbalance in class. One of the parts of collected dataset is presented in Fig 4. It shows the images of class showing '1'.

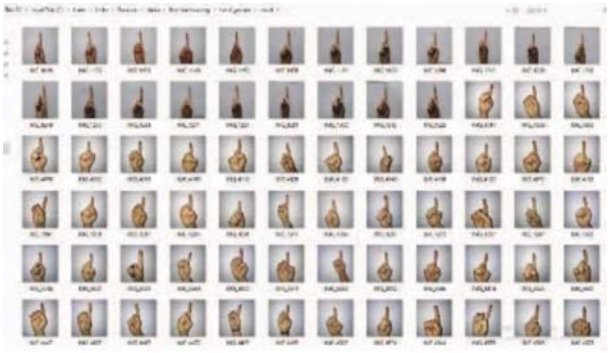


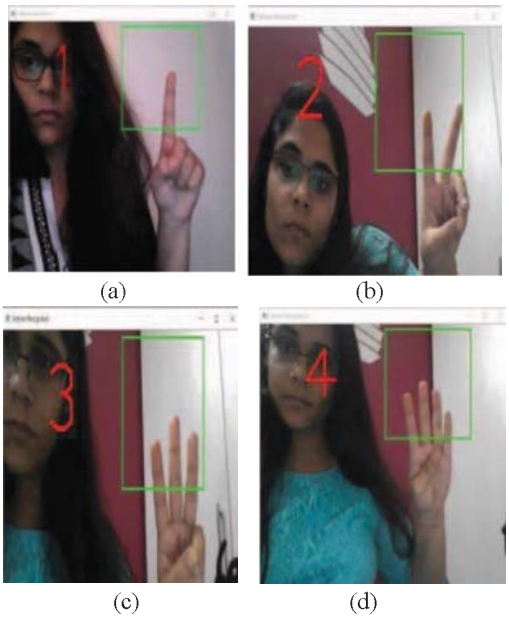
Fig. 4. Dataset of one of the labels

All the necessary libraries like Keras, TensorFlow, etc. were imported and data set was gathered using OpenCV library followed by an implementation of data augmentation technique which is an approach allows experts to expressively rise the variety of data available for model training, without collecting much data.

V. RESULT

The experimental results of this paper show that the model proposed in this paper can distinguish among several dominant and low-level features for the input images and can classify various hand gestures [8] with greater accuracy and a negligible model loss of 0.0504.

Fig 5(a) illustrates the hand gestures showing the value "1", 5(b) represents "2", 5(c) represents 3", 5(d) represents "4" and 5(e) represents "5" being captured and detected successfully.



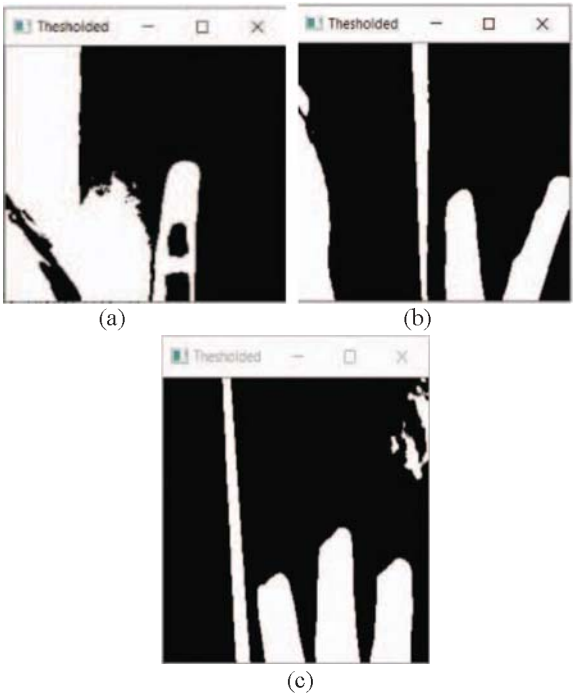
(a) (b)

(c) (d)

(e)

Fig 5. Hand gesture detection

Fig 6(a), 6(b), 6(c), 6(d) and 6(e) represents the pre-processed threshold images of "1", "2", "3", "4" and "5" respectively.



(a) (b)

(c)

(d) (e)

Fig 6. Threshold images

Before actual detection, the images to be trained are segmented. Afterwards, features [9] are extracted and reduced (finding the significant features and removing the unnecessary details that may create noise or reduce accuracy). Fig 7 presents a graph of results received from the presented model.

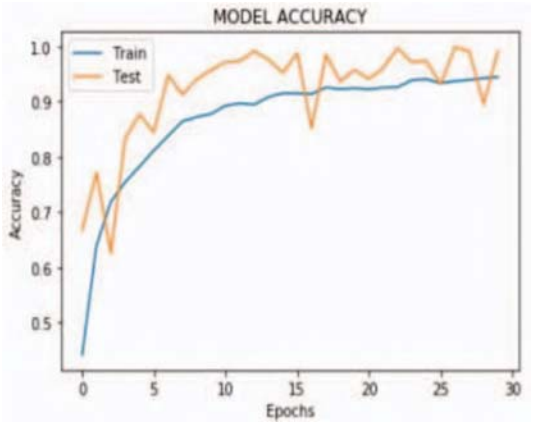


Fig 7. Result accuracy graph of the model

The model was able to achieve a testing accuracy of 99.13% which was much higher than the existing models. Gaining an accuracy as high as that was a tough job but it can be achieved by tuning various hyperparameters [10] and by performing proper data pre-processing and augmentation techniques.

Data pre-processing and data augmentation values:

Rescaling = 1./255
Rotation range = 20
Width shift range = 0.2
Height shift range = 0.2
Horizontal flip = True

The techniques like rescaling, cropping, padding, and horizontal flipping are very common for training dense neural networks. Data pre-processing [11] is also an important step which is mostly used in cases where we need to reduce the amount of data. It reduces the number of attributes, number of attribute values and the number of tuples. The most optimal values upon testing the accuracy of model [12] were as mentioned in the results above.

Here, Table 1 represents the values of various parameters that were tuned. The model was tested and the validation score of the model is calculated. The resultant Deep Convolution Network model comprises of seven hidden layers and tuned Hyperparameters [14] for our application.

Table 1. Optimal hyperparameters for the CNN

Parameters	Value
Hidden Layers	7
Dropout layer	1
Number of filters per Convolutional layer	32,64
Nodes in Dense layer	128
Batch size	32
Activation Function	ReLU
Optimizer	Adam
Epochs	30
Kernel Size	(3, 3)
Size of Pool	(2, 2)
Strides	(2, 2)

VI. CONCLUSION AND FUTURE WORK

This paper discusses and offers a state-of-the-art deep Multi-layer Convolutional Neural Network for performing hand gesture recognition in Human-Robot Interaction systems. It is an efficient model to be used on image data when tuned properly and with proper image pre-processing [15]. While detecting the images on live-videos or static images, the images and labels that were fed and trained in the model are used to compare the output. Its palpable ability to determine the invariant problem of recognizing gestures despite all the noise and complications is undefeatable. Also, it has been posed with real hand gesture images and despite hefty numbers and boundless structural overlapping of noises [16], the program worked fine along with providing a decent accuracy notch. This work can further be extended by adding more functionalities to the model and by making a greater number of classes.

Above mentioned explications put it in persuasive distinction to propose the mechanisms of recognition [17] which only work on meek images having fewer varying objects to be distinguished [18]. Therefore, this work is afoot in a door. Its remaining complications are meant to be resolved progressively.

REFERENCES

[1] Pigou, Lionel, et al. "Sign language recognition using convolutional neural networks." *European Conference on Computer Vision*. Springer, Cham, pp. 572-578, 2014.

[2] Li, Gongfa, et al. "Hand gesture recognition based on convolution neural network." Cluster Computing pp.2719-2729, 2017.
[3] Nagi, Jawad, et al. "Max-pooling convolutional neural networks for vision-based hand gesture recognition." 2011 IEEE International Conference on Signal and Image Processing Applications (ICSIPA). IEEE, pp. 342-347, 2011.
[4] Molchanov, Pavlo, et al. "Hand gesture recognition with 3D convolutional neural networks." Proceedings of the IEEE conference on computer vision and pattern recognition workshops. pp. 1-7, 2015.
[5] Molchanov, Pavlo, et al. "Online detection and classification of dynamic hand gestures with recurrent 3d convolutional neural network." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4207-4215, 2016.
[6] Tolias, Giorgos, Ronan Sicre, and Hervé Jégou. "Particular object retrieval with integral max-pooling of CNN activations." *arXiv preprint arXiv:1511.05879,* 2015.
[7] Ren, Zhou, et al. "Robust part-based hand gesture recognition using kinect sensor." IEEE transactions on multimedia 15.5, pp.1110-1120, 2013.
[8] Nishihara, H. Keith, et al. "Hand-gesture recognition method." U.S. Patent No. 9,696,808. 4 Jul, 2017.
[9] Chaudhary, Anita, and Sonit Sukhraj Singh. "Lung cancer detection on CT images by using image processing." *2012 International Conference on Computing Sciences*. IEEE, pp. 142-146, 2012.
[10] Binh, Nguyen Dang, Enokida Shuichi, and Toshiaki Ejima. "Real-time hand tracking and gesture recognition system." *Proc. GVIP* pp.19-21, 2005.
[11] Murthy, G. R. S., and R. S. Jadon. "A review ofvision based hand gestures recognition." *International Journal of Information Technology and Knowledge Management* 2.2, pp.405-410, 2009.
[12] Manresa, Cristina, et al. "Hand tracking and gesture recognition for human-computer interaction." ELCVIA Electronic Letters on Computer Vision and Image Analysis 5.3, pp.96-104, 2005.
[13] Agarap, Abien Fred. "Deep learning using rectified linear units (relu)." *arXiv preprint arXiv:1803.08375* 2018.
[14] Wang, Binghui, and Neil Zhenqiang Gong. "Stealing hyperparameters in machine learning." *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE, pp. 36-52, 2018.
[15] Poostchi, Mahdieh, et al. "Image analysis and machine learning for detecting malaria." *Translational Research*194, pp.36-55, 2018.
[16] Santhanam, T., and S. Radhika. "A Novel Approach to Classify Noises in Images Using Artificial Neural Network1." 2010.
[17] Simonyan, Karen, and Andrew Zisserman. "Verydeep convolutional networks for large-scale image recognition." *arXiv preprint arXiv:1409.1556* 2014.
[18] Kim, Youngwook, and Brian Toomajian. "Hand gesture recognition using micro-Doppler signatures with convolutional neural network." *IEEE Access* 4,pp.7125-7130 2016.