VERSATILE SIGN LANGUAGE COMMUNICATOR

Arunima Rashmi Giri[1], Sanmayee Mahabaleshwar Hegde[2], T S Kushal[3], Ashritha R Murthy[4]
B.E in Computer Science Department, JSS Science and Technology University. Mysore,

Abstract: The system designed here hopes to enable simple and e cient communication between individuals who use sign language to communicate and those who do not have any knowledge of sign language. It is designed so that when a user makes gestures they will be translated in real time to their corresponding text and audio formats. The system will also try to enable a speech to text functionality, which will allow users to communicate with hearing impaired individuals even if they do not know any sign languages. We hope to provide a platform where people located near each other will be able to communicate e ectively with each other on a single device even if one of them is hearing impaired and the other does not have any knowledge of sign language. Finally, the system will also attempt to provide a way for people who wish to communicate and are physically far away to communicate with the above like interface over the internet.

I. INTRODUCTION

1.1 Aim/statement of the problem

To build a software which converts sign language into speech and provides var-ious means of communication facilitating sign language.

1.2 Objectives of the project work

To allow for translation from sign language to text to speech and vice versa.
To allow for real-time conversion from Indian Sign Language to American Sign Language.
To allow for real-time video communication with Sign Language sup-ported.

1.3 Introduction to the problem domain

Sign languages (also known as signed languages) are languages that use the visual-manual modality to convey meaning. Sign languages are expressed through manual articulations in combination with non-manual elements. Sign languages are full- edged natural languages with their own grammar and lexicon.[2] Sign languages are not universal and they are not mutually intelligible with each other, although there are also striking similarities among sign languages. Linguists consider both spoken and signed communication to be types of natu-ral language, meaning that both emerged through an abstract, protracted ag-ing process and evolved over time without meticulous planning. Sign language should not be confused with body language, a type of nonverbal communication.

Sign Language consists of the following 3 major components:

Fingerspelling	Word level sign vocabulary	Non-manual features
Used to spell words letter by letter .	Used for the majority of communication.	Facial expressions and tongue, mouth and body position.

Figure 1.1: 3 Major Components of Sign Language

In our project we basically focus on producing a model which can recognise Finger-spelling based hand gestures in order to form a complete word by com-bining each gesture. The gestures we aim to train are as given in the image below.[1]

Figure 1.2: Hand Signs in American Sign Language (ASL)

Wherever communities of deaf people exist, sign languages have developed as handy means of communication and they form the core of local deaf cultures. Although signing is used primarily by the deaf and hard of hearing, it is also used by hearing individuals, such as those unable to physically speak, those who have trouble with spoken language due to a disability or condition (augmenta-tive and alternative communication), or those with deaf family members, such as children of deaf adults.
Our aim here is to create a system which will allow speech and hearing impaired people and those who don't know sign language to communicate e ectively with each other. The system rst will attempt to translate sign language gestures into text and audio. Then will move onto allow for conversion from speech to text. Finally, we hope to provide a system that will allow users who are remotely located to

communicate e ciently with each other.

1.4 Applications

Allow for easy communication between people who know sign language and those who do not.
Ease of communication between people through speech output for the translated gestures.
Allow for communication between people who are physically near each other and one or both of them is speech or hearing impaired.
Communication between people who are located remotely i.e. far away from each other over the internet.

1.5 Existing solution methods

There are sign language to text and text to speech software which have been developed. We hope to expand on this domain and try to combine the two techniques and have a sign language to text to speech software. Speech to text software have also been developed, we hope to incorporate this in an e cient manner in our system. So far there have been no proposed solution for the problem of communication between people through sign language over a distance where one of the individuals does not know sign language.

1.6 Proposed solution methods

We plan to implement a system that will allow for translation from sign language to text to speech in real time and also allow for speech to text conversion in the same system. This would solve the problem of two people communicating who are located near to each other. For people who wish to communicate over distances we plan to implement a module that will allow two users to communicate over the internet via video call with the translation from sign language to text to speech and speech to text happening simultaneously in real time with minimum lag. We hope this will allow for increased ease of communication between hearing impaired individual and those who do not know sign language.

1.7 Time schedule for completion of the project work (Gantt chart)

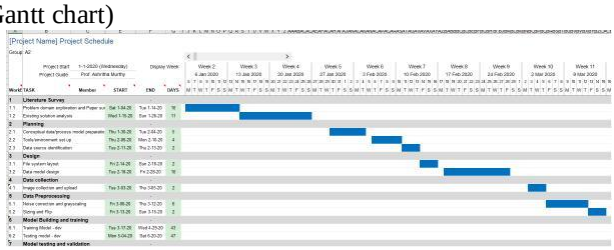

Figure 1.3: Gantt Chart 1st Part

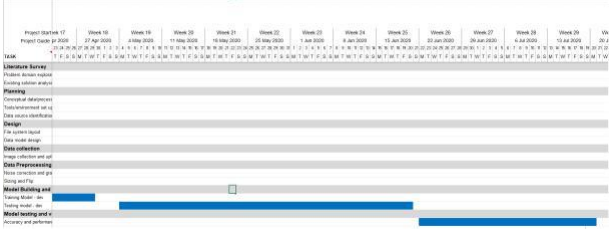

Figure 1.4: Gantt Chart 2nd Part

II. LITERATURE SURVEY

In the recent years there has been tremendous research done on the hand gesture recognition.[4] With the help of literature survey done we realized the basic steps in hand gesture recognition are:
Data acquisition
Data preprocessing Feature extraction Gesture classi cation

2.1 Data acquisition

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

2.1.1 Use of Sensory Devices

It uses electro-mechanical devices to provide exact hand con guration, and po-sition. Di erent glove based approaches can be used to extract information. But it is expensive and not user friendly.

2.1.2 Vision based approach

In vision based methods computer camera is the input device for observing the information of hands or ngers. The Vision Based methods require only a cam-era, thus realizing a natural interaction between humans and computers without the use of any extra devices. These systems tend to complement biological vi-sion by describing 7 arti cial vision systems that are implemented in software and/or hardware. The main challenge of vision-based hand detection is to cope with the large variability of human hand's appearance due to a huge number of hand movements, to di erent skin-color possibilities as well as to the variations in viewpoints, scales, and speed of the camera capturing the scene.

2.2 Data preprocessing and Feature extraction for vision based approach

The approach for hand detection combines threshold-based color detec-tion with background subtraction. We can use Adaboost face detector to di erentiate between faces and hands as both involve similar skin-color.
We can also extract necessary image which is to be trained by applying a lter called Gaussian blur. The lter can be easily applied using open computer vision also known as OpenCV.
For extracting necessary image which is to be trained we can use instru-mented gloves. This helps reduce computation time for preprocessing and can give us more concise and accurate data compared to applying lters on data received from video extraction.
We tried doing the hand segmentation of an image using color segmenta-tion techniques but as mentioned in the research paper skin color and tone is highly dependent on the lighting conditions due to which output we got for the segmentation we tried to do were no so great. Moreover we have a huge number of symbols to be trained for our project many of which look similar to each other like the gesture for symbol 'V' and digit '2', hence we decided that in order to produce better accuracies for our large number of symbols, rather than segmenting the hand out of a random background we keep background of hand a stable single color so that we don't need to segment it on the basis of skin color. This would help us to get better results.

2.3 Gesture classi cation

Hidden Markov Models (HMM) is used for the classi cation of the ges-tures .This model deals with dynamic aspects of gestures. Gestures are extracted from a sequence of video images by tracking the skin-color blobs corresponding to the hand into a body{ face space centered on the face of the user. The goal is to recognize two classes of gestures: deictic and symbolic.[3] The image is ltered using a fast look{up indexing table. Af-ter ltering, skin color pixels are gathered into blobs. Blobs are statistical objects based on the location (x, y) and the colorimetry (Y, U, V) of the skin color pixels in order to determine homogeneous areas.

Naive Bayes Classi er is used which is an e ective and fast method for static hand gesture recognition. It is based on classifying the di erent gestures according to geometric based invariants which are obtained from image data after segmentation. Thus, unlike many other recognition meth-ods, this method is not dependent on skin color. The gestures are ex-tracted from each frame of the video, with a static background. The rst step is to segment and label the objects of interest and to extract geo-metric invariants from them. Next step is the classi cation of gestures by using a K nearest neighbor algorithm aided with distance weighting al-gorithm (KNNDW) to provide suitable data for a locally weighted Naive Bayes classi er.

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

2.3.1 Arti cial Neural Networks

Arti cial Neural Network is a connections of neurons, replicating the structure of human brain. Each connection of neuron transfers information to another neuron. Inputs are fed into rst layer of neurons which processes it and trans-fers to another layer of neurons called as hidden layers.[6] After processing of information through multiple layers of hidden layers, information is passed to nal output layer. There are capable of learning and they have to be trained.

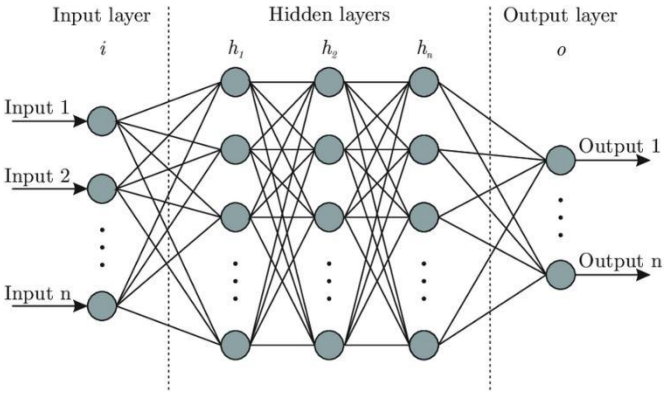

Figure 2.1: Basic Architecture of Arti cial Neural Networks

There are di erent learning strategies:

- Unsupervised Learning
- Supervised Learning
- Reinforcement Learning

2.3.2 Convolution Neural Network

Unlike regular Neural Networks, in the layers of CNN, the neurons are arranged in 3 dimensions: width, height, depth. The neurons in a layer will only be connected to a small region of the layer (window size) before it, instead of all of the neurons in a fully-connected manner. Moreover, the nal output layer would have dimensions (number of classes), because by the end of the CNN architecture we will reduce the full image into a single vector of class scores.[9] The layers are of the following types:

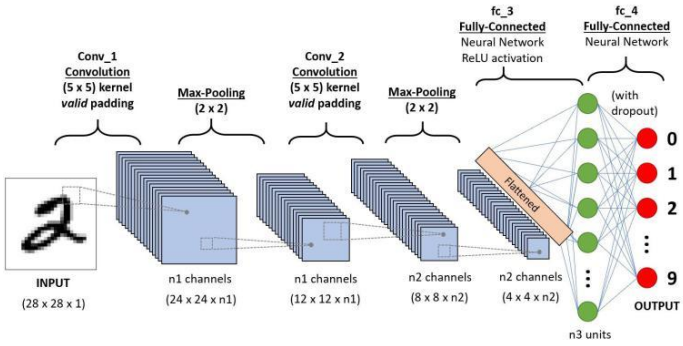

Figure 2.2: Layers in Convolution Neural Networks

Convolution Layer

In convolution layer we take a small window size [typically of length 5*5] that extends to the depth of the input matrix. The layer consist of learn-able lters of window size. During every iteration we slid the window by stride size [typically 1], and compute the dot product of lter entries and input values at a given position. As we continue this process well create a 2-Dimensional activation matrix that gives the response of that matrix at every spatial position. That is, the network will learn lters that activate when they see some type of visual feature such as an edge of some orientation or a blotch of some color.

Pooling Layer

We use pooling layer to decrease the size of activation matrix and ulti-mately reduce the learn-able parameters.

There are two type of pooling:
Max Pooling
In max pooling we take a window size [for example window of size 2*2], and only take the maximum of 4 values. Well lid this window and continue this process, so well nally get an activation matrix half of its original Size.

Average Pooling
In average pooling we take average of all values in a window.

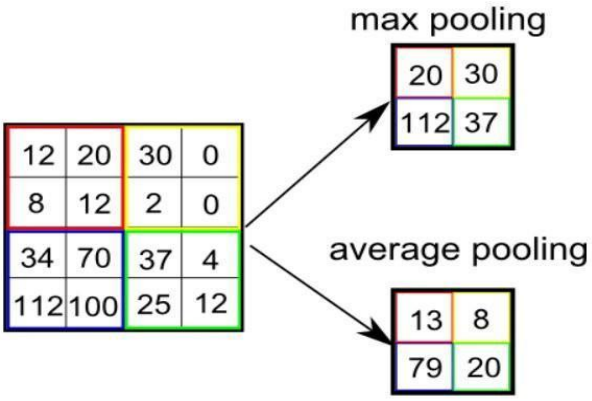

Figure 2.3: Types of Pooling in CNN

Fully Connected Layer
In convolution layer neurons are connected only to a local region, while in a fully connected region, well connect the all the inputs to neurons.

Final Output Layer
After getting values from fully connected layer, well connect them to nal layer of neurons (having count equal to total number of classes), that will predict the probability of each image to be in di erent classes.

Basically there are two approaches for sign recognition vision based and sen-sor based gesture recognition. Lots of study has been done on sensor based approaches like gloves, wires, helmets etc. but due to disadvantage of wear it continuously is not possible, therefore further work is concentrated on Image based approaches.
Some earlier work has done on image based approaches for hand gesture and sign recognition in last decade. There have been various approaches for gesture

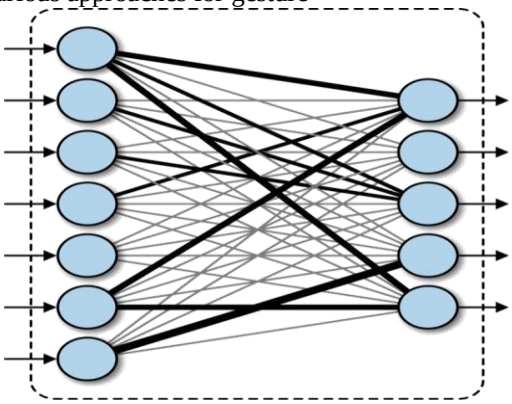
Figure 2.4: Basic Fully Connected Layer

recognition like HMM (Hidden markov model), ANN (Arti cial neural network).

Eigen value based, perceptual color based.
Iwan Njoto Sandjaja and Nelson Marcos proposed color gloves approach which extracts important features from the video using multi-color tracking algorithm. Ibraheem and Khan have reviewed various techniques for gesture recognition and recent gesture recognition approaches. Ghotkar et al used Cam shift method and Hue, Saturation, Intensity (HSV) color for model for hand tracking and segmentation. For gesture recognition Genetic Algorithm is used.

Methods like Support Vector Machines (SVM) proposed for classi cation and particle ltering. There are various methods for image segmentation. The HSV, color space de nes color with intuitive values.[5] The Perception of the color space components and Discrimination between luminance and chromi-nance components works e ectively for skin color segmentation. Therefore HSV color model selected for hand Segmentation.

The various methods used in Feature Extraction [ANN] Describes those meth-ods: Contour shape techniques which extract boundary information of signs. One of the most prominent achievements that were achieved in the eld of sign to voice conversion and voice to sign which can remove communication barrier.

Image based gesture recognition system is divided into three steps In Image-preprocessing color to binary conversion and Noise ltering is done for captured image. Tracking is mainly used for tracking a hand gesture from capture im-age using Convexity hull algorithm. Finally recognition is done with the help of features like convex hull and convex defects taken from tracking.[6] Image based gesture recognition can be used in many applications one of them is sign language recognition which is as explained below.

III. SYSTEM REQUIREMENTS AND ANALYSIS

3.1 System Overview

The system has been designed so as to provide the users with easy access to a translation platform which converts the sign language into the corresponding text. The main aim of the system is to facilitate ease of communication between speech and hearing impaired community and the speaking community. The sys-tem will attempt to provide this communication platform between people who are physically near each other and also those who are attempting to communi-cate from remote locations.[7] First, the signs made by the user are captured by a camera and then this image is used for comparison with pre-stored binary images of all the signs in the sign language, allowing for identi cation of the sign. Its corresponding text i.e. meaning is read from a database and then the text is shown in a window beside the user's camera window. This translation from sign to text is being done in real-time, with minimum lad time, to

allow for faster and simpler communication. Further, the text so obtained will be feed into a text-to speech converter so that the translation is read out loud by the system as soon as the user performs the related gesture. Next, the system will also attempt to translate speech to text so that a person who doesn't know sign language can speak into a microphone and the corresponding text will be dis-played on the screen. Thus this will allow a person with a hearing impairment and a person who doesn't know sign language to communicate e ectively. We hope that with the ease of communication being provided here we can manage to bridge the gap between people everywhere.

The above is the rst phase of the project. Phase 1 will allow for commu-nication between people who are physically present near each other. In Phase 2 we plan to address the problem of communication between people who are attempting to communicate remotely over the internet. For this we hope to provide an IOT solution where two users will be able to communicate with each other over a video call like interface. As such the two users will each have a window open on their respective devices and be able to see each other, as in the usual case of a video call, with the additional capability that any sign language gesture made by either of the users will be converted to text and displayed on the other's screen and will also broadcast the audio equivalent of the text over the user's speakers. The system will also attempt to create an alternate inter-face where the users will only have to be connected over a regular voice call. Here the speech impaired user will make the sign language gestures in-front of their camera and this will be converted and sent as text and audio to the other user. The other user will be able to speak directly into their phone and have the text appear on their companion's device. We hope to facilitate this commu-nication in a cost e ective way over the internet and also provide regular o ine capability for when the communicating users are physically near each other.

3.2 Functional Requirements

The system should be able to support translation from sign language to text and audio format. It should be able to support this translation in real-time with a minimum time lag. It should support recognition of various skin tones and hand sizes. It should also support gestures recognition for gestures made by either the right or the left hand. The system should be able to recognize all the gestures made within a certain time-frame.[6] The text i.e. translation of the signs should be displayed in a prominent format, such that it is easily read-able and understandable by any individual. The audio produced from the text should also be correspondingly audible and the user should be given the option to mute it if needed. Webcam support has to provided, so that the users can use the system directly from their laptops. The system should also support minor changes in the camera resolution available to it in the user's environment. The system should be easily deployed, so that a user without any technical knowl-edge can start and use it without any trouble. For this, the system should be started directly from an executable le (here batch le), so that the internal commands are executed without the need for user intervention. This will allow the individuals to use the system by the simple click of a button.

The speech to text module should also be user friendly. It should support recognition from any microphone which is available to the user. The user has to receive clear instructions on how to activate the speech to text software and it should be made easy to access and use, maybe with the execution of a single le. While speaking the system should ignore the ambient noise emanating from the user's surroundings. The system should also be able to recognize the voice of the user up to a certain volume level and should inform the speaker to speak more loudly when the words being spoken cannot be recognized. The system should also support minor changes in the microphone quality available to it in the user's environment. The system should be easily deployed, so that a user without any technical knowledge can start and use it without any trouble. For this, the system should be started directly from an executable le (here batch le), so that the internal commands are executed without the need for user in-tervention. Installation of the system on a user's device should also be simple and e cient. The installation of the system should not a ect other programs running on a user's device. It should also be such that the user can install the system without any knowledge about the technologies being used.

The system should be available with a simple user interface with the user's camera being used capture the gestures being made in sign language and the text being displayed in another window just beside the capture window. The text should also be displayed in real time in the window in an easily readable format. The system should also access the user's speakers to give the audio output from the text to speech software at a suitable level, such that the words are clearly heard by the user. For the speech to text module the system should be able to access the microphone connected to the user's system and obtain the raw data from there to convert into text while avoiding excess noise from the user's surroundings. The system should also display the converted text in another window to provide a clear understanding of where the data is coming from i.e. the camera (Sign Language to Text) or the microphone (Speech to Text).

3.3 Non-Functional Requirements

Performance The system should be able to convert all the gestures made by the user within a certain time-frame without any lag. It should be able to support translation of sign language gestures which are made with a certain degree of speed. The speech to text module should be able to handle heavy load as the user may speak several words per minute. Under heavy load or usage the system should not crash and should function without much loss of performance.
Security The conversations happening between the users should not be recorded in any manner. The users should have the option to disable certain modules, so that they can disable their webcam or microphone under di erent circumstances. Once the software is shut down there should not be any access to the user's webcam or their microphone from the

system.
Usability The system should be user friendly. It should be able to start the system and start using it with the simple execution of a single program. The user should be able to use the system even if they have no knowledge of the technical aspects of the underlying software. A step-by-step guide should be provide so that the user can refer to it on how to best utilize the system. The installation should also be a simple process and the user should be able to do it in an e cient and cost e ective manner. O ine mode should be provided for when the individuals communicating are physically near to each other.

Reliability The system should be reliable such that it should provide the correct meanings of the gestures in sign language being made by the user. It shouldn't translate the sign language gestures incorrectly and should raise an error when a certain gesture is not found in the database. The system should function correctly in any environment that the user deploys it. It should not crash when it is overloaded with information but should raise an error that the data cannot be processed. The audio output should be reliable and should not have too much noise and should be easily discernable by the user.

IV. TOOLS AND TECHNOLOGY USED

4.1 Python Libraries Used

4.1.1 NumPy

NumPy is the fundamental package for scienti c computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.
At the core of the NumPy package, is the ndarray object. This encapsulates n-dimensional arrays of homogeneous data types, with many operations being performed in compiled code for performance. There are several important dif-ferences between NumPy arrays and the standard Python sequences:
NumPy arrays have a xed size at creation, unlike Python lists (which can grow dynamically). Changing the size of an ndarray will create a new array and delete the original.
The elements in a NumPy array are all required to be of the same data type, and thus will be the same size in memory. The exception: one can have arrays of (Python, including NumPy) objects, thereby allowing for arrays of di erent sized elements.
NumPy arrays facilitate advanced mathematical and other types of oper-ations on large numbers of data. Typically, such operations are executed more e ciently and with less code than is possible using Pythons built-in sequences.
A growing plethora of scienti c and mathematical Python-based packages are using NumPy arrays; though these typically support Python-sequence input, they convert such input to NumPy arrays prior to processing, and they often output NumPy arrays.
We have used numpy to facilitate the easy access and manipulation of the data, since we're dealing with a large number of images which are usually represent as matrices.

4.1.2 Scikit-learn or sklearn

Scikit-learn is probably the most useful library for machine learning in Python. It is on NumPy, SciPy and matplotlib, this library contains a lot of e cient tools for machine learning and statistical modeling including classi cation, re-gression, and clustering and dimensionality reduction.
Scikit-learn is largely written in Python, and uses numpy extensively for high-performance linear algebra and array operations. Scikit-learn integrates well with many other Python libraries, such as matplotlib and plotly for plotting, numpy for array vectorization, pandas dataframes, scipy, and many more.
We used this technology in the beginning for training the model but later ob-served that the results obtained were not compatible with our expected out and hence we shifted to other technologies.

4.1.3 OpenCV

OpenCV(Open Source Computer Vision) is an open source library of program-ming functions used for real-time computer-vision. It is mainly used for image processing, video capture and analysis for features like face and object recogni-tion. It is written in C++ which is its primary interface, however bindings are available for Python, Java, MATLAB/OCTAVE.
We decided to use this technology because it allows for easy and e cient ma-nipulation of images. It also facilitates easy capture of images from a continual video stream and even helps in writing back on the images any required text, which is needed in our system. The software uses OpenCV to start the user's video camera and then create a binary thresholding window to allow for the capture of the hand's histogram, which is then used to eliminate noise as the model is trained to recognize skin tone and the shape of the hand. OpenCV is also used to capture the images and rotate them so that the model can recognize the user's hand from multiple angles. Finally, it's used to close all the relevant windows so that the user's webcam is not is not used injudiciously.

4.1.4 Keras

Keras is an open-source neural-network library written in Python. It is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, R, Theano, or PlaidML. Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible. It o ers a higher-level, more intuitive set of abstractions that make it easy to develop deep learning models regardless of the computational back-end used. Keras contains numer-ous implementations of commonly used neural-network building blocks such as layers, objectives, activation functions, optimizers, and a host of tools to make working with image and text data easier to simplify the coding necessary for writing deep neural network code. In addition to standard neural networks, Keras has support for

convolutional and recurrent neural networks.

This was a necessary requirement for us to train our data-set, as CNN is our model of choice. Hence the CNN was implemented with the help of Keras.

4.1.5 TensorFlow

TensorFlow is an open source software library for numerical computation using data ow graphs. Nodes in the graph represent mathematical operations, while the graph edges represent the multidimensional data arrays (tensors) communi-cated between them. The exible architecture allows you to deploy computation to one or more CPUs or GPUs in a desktop, server, or mobile device with a single API. It is used for machine learning applications such as neural networks. It is used for both research and production at Google, often replacing its closed-source predecessor, DistBelief.

TensorFlow computations are expressed as stateful data- ow graphs. The name TensorFlow derives from the operations that such neural networks perform on multidimensional data arrays. These arrays are referred to as "tensors".

TensorFlow separates the de nition of computations from their execution even further by having them happen in separate places: a graph de nes the oper-ations, but the operations only happen within a session. Graphs and sessions are created independently. A graph is like a blueprint, and a session is like a construction site.

To do e cient numerical computing in Python, we typically use libraries like NumPy that do expensive operations such as matrix multiplication outside Python, using highly e cient code implemented in another language. Unfortu-nately, there can still be a lot of overhead from switching back to Python every operation. This overhead is especially bad if you want to run computations on GPUs or in a distributed manner, where there can be a high cost to transferring data. TensorFlow also does its heavy lifting outside Python, but it takes things a step further to avoid this overhead. Instead of running a single expensive operation independently from Python, TensorFlow lets us describe a graph of interacting operations that run entirely outside Python. This approach is simi-lar to that used in Theano or Torch.

It is a symbolic math library, and is also used for machine learning applica-tions such as neural networks. TensorFlow manipulates data by creating a Data-Flow graph or a Computational graph. It consists of nodes and edges that perform operations and do manipulations like addition, subtraction, multipli-cation, etc. TensorFlow is now being widely used to build complicated Deep Learning models.

4.1.6 pyttsx3

pyttsx3 is a text-to-speech conversion library in Python. Unlike alternative libraries, it works o ine and is compatible with both Python 2 and 3. An application invokes the pyttsx3.init() factory function to get a reference to a pyttsx3. Engine instance. it is a very easy to use tool which converts the en-tered text into speech. The pyttsx3 module supports two voices rst is female and the second is male which is provided by \sapi5" for windows.

It supports three TTS engines:

sapi5 { SAPI5 on Windows

nsss { NSSpeechSynthesizer on Mac OS X espeak { eSpeak on every other platform

4.1.7 Tkinter

Python o ers multiple options for developing GUI (Graphical User Interface). Out of all the GUI methods, tkinter is the most commonly used method. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with Tkinter is the fastest and easiest way to create the GUI applications. Cre-ating a GUI using tkinter is an easy task.

To create a Tkinter app:

- Importing the module { tkinter
- Create the main window (container)
- Add any number of widgets to the main window
- Apply the event Trigger on the widgets

There are two main methods used which the user needs to remember while creating the Python application with GUI.

Tk(screenName=None, baseName=None, className='Tk', useTk=1): To create a main window, tkinter o ers a method 'Tk(screenName=None, baseName=None, className='Tk', useTk=1)'. To change the name of the window, you can change the className to the desired one.

mainloop():

There is a method known by the name mainloop() is used when your application is ready to run. mainloop() is an in nite loop used to run the application, wait for an event to occur and process the event as long as the window is not closed.

Tkinter also o ers access to the geometric con guration of the widgets which can organize the widgets in the parent windows. There are mainly three geom-etry manager classes class.

pack() method: It organizes the widgets in blocks before placing in the parent widget.

grid() method: It organizes the widgets in grid (table-like structure) before placing in the parent widget.

place() method: It organizes the widgets by placing them on speci c positions directed by the programmer.

4.2 Data-Set

Figure 4.1: Signs Captured and Stored to be used as the Data Set

The data-set is made up of the hand signs representing the various alphabets. These hand signs are saved as pictures depicting the outline of the hand with which the gesture is being made. The images are obtained after capturing a picture of the hand and then applying a gaussian lter to obtain the nal images which are used to train the model. Multiple images of each hand sign have been captured to be used for the training and testing of the system.

V. SYSTEM DESIGN

The most basic functionality of the software is to recognize the hand gestures made by the user. For this, the software is designed to make use of video camera facility provided by openCV technology which has ability to detect the human skin color.

5.1 Development perspective

In the beginning, system requires a set of images to be fed into it which is later used to predict the textual form of the gesture. For this reason, we have trained the system with possible gestures. As and when a gesture is performed, the software keeps on capturing the images until it reaches a prede ned count which is required to obtain a more reliable prediction. To keep track of the ges-tures uniquely, system demands for the gesture id and gesture name. To store these data we have made use of SQLite database, while the images gets stored in a folder. If there is a con ict in gesture id, system displays a warning asking whether to replace the existing gesture and the action is performed accordingly. Once the images are obtained, each image is picked and is ipped so as to allow the prediction of gestures made by both the hands. Eighty percent of the images obtained in total are used for training the system and remaining are used for the testing phase.

We decided on using Convolution Neural Networks (CNN) because it is a stan-dard extension of deep learning and it also e ectively reduces the number of parameters needed for the neural net. Di erent technologies were then used to train the model and then the results were analyzed and it was found that sklearn didn't produce outputs which corresponded to our expected results. We then found that training with Keras produced the desired outputs.

Then, a histogram of the hand was obtained by capturing images from the laptop's webcam. The histogram is then obtained from the captured image by thresholding at a certain value and then applying suitable transformations to obtain the correct form for the histogram. This histogram is used to identify the user's hand and avoid the ambient noise that is present in the captured images.

5.2 User perspective

To allow the user to perform the gesture, a pop up video camera is provided. This camera interface keeps predicting the possible textual form of the gesture until it is able to completely and accurately recognize the gesture. At the present, system has ability to recognize and predict the gesture to serve one user. We are hoping to equip the system design with one to one interface which will dynamically take input from one side and displays the predicted text on both the sides which is required to maintain the accuracy of communication.

VI. SYSTEM IMPLEMENTATION

6.1 Methodology

The system is a vision based approach. All the signs are represented with bare hands and so it eliminates the problem of using any arti cial devices for inter-action. The problem with other methods such as sensory gloves etc. are that external equipment is required for each use that is not feasible to deploy for a large number of users. We used Convolution Neural Networks (CNN) to train the model to identify the gestures made by the user. First, we capture the his-togram of a hand so as to allow the system to recognize the user's hand when the user makes gestures in front of the video camera or webcam of the user's laptop. Then, di erent hand gestures were recorded in a database and then they were rotated to make the gestures identi able from any angle. Lastly, the model was trained with the recorded images using Keras and then tested with real-time gestures.

The process can be summarized in the following steps:

- Set hand histogram
- Create gestures
- Flip images
- Load images
- Display all gestures
- Train the model

The above steps have been followed so far till the training step with di erent methods and optimizations being applied to the training of the system.

6.2 Data Set Generation

For the project we tried to nd already made data-sets but we couldn't nd data-set in the form of raw images that matched our requirements. All we could nd were the data-sets in the form of RGB values. Hence we decided to create our own data set. Steps we followed to create our data set are as follows.

We used Open computer vision (OpenCV) library in order to produce our data-set.[9] Firstly we captured around 800 images of each of the symbol in ASL for training purposes and around 200 images per symbol for testing purpose. First we capture each frame shown by the webcam of our machine. In the each frame we de ne a region of interest (ROI) which is denoted by a blue bounded square as shown in the image below. From this whole image we extract our ROI which is RGB and convert it into gray scale Image as shown below.

Finally we apply our Gaussian blur lter to our image which helps us extracting various features of our image. The image after applying Gaussian blur looks like below.

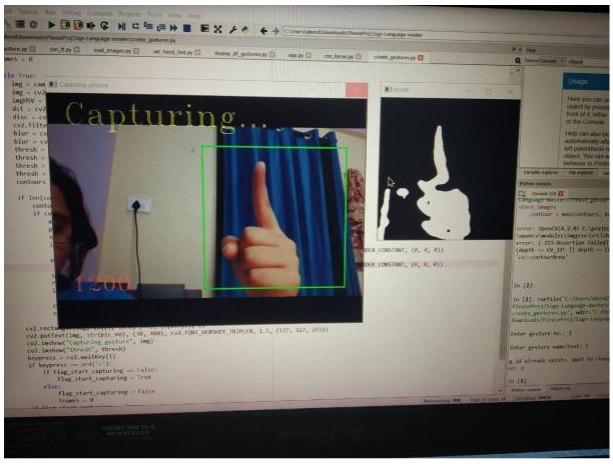

Figure 6.1: Capturing a Hand Sign and applying a Gaussian Filter

6.3 Gesture Classi cation

We have used convolution neural networks for the training of our model.

The following steps were followed for this:

- We created a CNN using both Tensor ow and Keras.
- Then we trained the model using Keras on a video stream.

Our approach uses two layers of algorithm to predict the nal symbol of the user.

6.3.1 Algorithm Steps in Layer 1

- Apply Gaussian blur lter and threshold to the frame taken with openCV to get the processed image after feature extraction.
- This processed image is passed to the CNN model for prediction and if a letter is detected for more than 50 frames then the letter is printed and taken into consideration for forming the word.
- Space between the words are considered using the blank symbol.

6.3.2 Algorithm Steps in Layer 2

- We detect various sets of symbols which show similar results on getting detected.
- We then classify between those sets using classi ers made for those sets only.

After training the model was able to recognize 36 characters i.e. 26 letters and 10 digits as shown below.

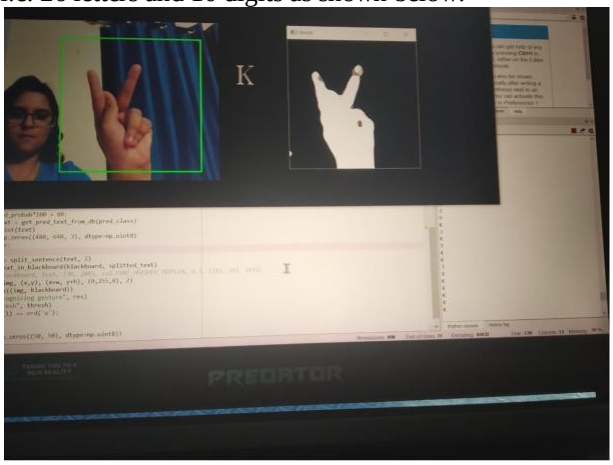

Figure 6.2: Recognizing the alphabet K

6.4 Finger Spelling Sentence Formation

- Whenever the count of a letter detected exceeds a speci c value and no other letter is close to it by a threshold we print the letter and add it to the current string(In our code we kept the value as 50 and di erence threshold as 20).
- Otherwise we clear the current dictionary which has the count of detec-tion of present symbol to avoid the probability of a wrong letter getting predicted.
- Whenever the count of a blank(plain background) detected exceeds a spe-ci c value and if the current bu er is empty no spaces are detected.
- In other case it predicts the end of word by printing a space and the current gets appended to the sentence below.

6.5 User Interface

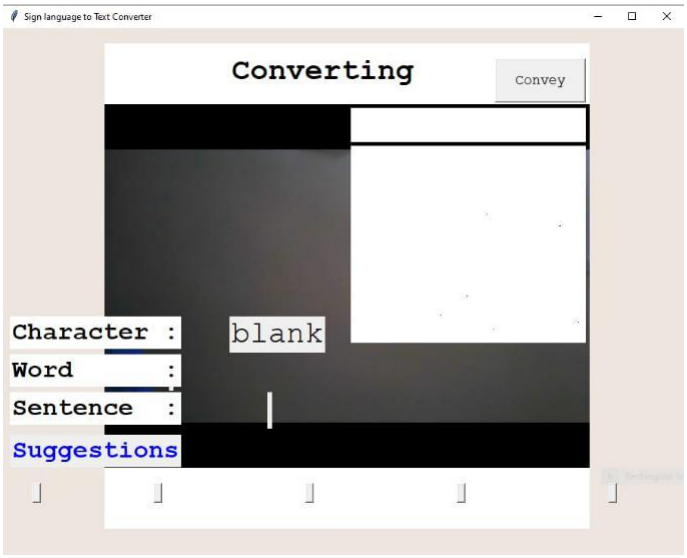

Figure 6.3: Screenshot of the User Interface

To enable easy interaction between the di erently abled, we have created a user friendly interface. This interface is created using python Tkinter. Tkinter is the standard Python interface to Tk GUI toolkit and is python's de facto standard GUI. The interface is named as Sign language converter. We have created a slot inside this interface which continuously shows the frames captured by webcam. This will enable the user to keep a track of the gesture being performed. Since our sign language converter uses nger-spelling technique, it is important that every letter's gesture being performed is visible to the user to verify its correctness.[5] Hence, the interface shows each character after a speci c number of frames. The word created by predicted letters is also shown. While the user is performing gestures, suitable suggestions are shown which act like auto correct feature and thus minimize the time required to predict the complete word. Finally, when the user is sure about the sentence that is to be conveyed, clicking on the button \Convey" will spell out the sentence.

VII. SYSTEM TESTING AND RESULTS ANALYSIS

7.1 Training and Testing

We convert our input images(RGB) into gray-scale and apply gaussian blur to remove unnecessary noise.We apply adaptive threshold to extract our hand from the background and resize our images to 128 x 128. We feed the input images after preprocessing to our model for training and testing after applying all the operations mentioned above. The prediction layer estimates how likely the im-age will fall under one of the classes. So the output is normalized between 0 and 1 and such that the sum of each values in each class sums to 1. We have achieved this using soft-max function.

At rst the output of the prediction layer will be somewhat far from the actual value. To make it better we have trained the networks using labeled data. The cross-entropy is a performance measurement used in the classi cation. It is a continuous function which is positive at values which is not same as labeled value and is zero exactly when it is equal to the labeled value. Therefore we optimized the cross-entropy by minimizing it as close to zero. To do this in our network layer we adjust the weights of our neural networks. TensorFlow has an inbuilt function to calculate the cross entropy.

7.2 Analysis

We then proceeded onto discovering trends in the accuracy changes against the test set. For testing purposes we created two test sets, one to set a base line test result and the other to validate the class ndings. Having two test sets makes sure that the classi cation done in the rst round on the test set isn't on chance in the condition that the second test result is coherent with the rst.[8]

We select the data in both sets using random distribution of all the 27 class labels. We observed that the model provided us with an accuracy of 96.02%

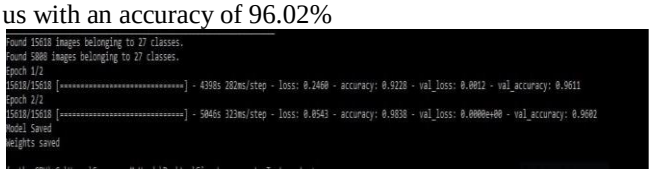

Figure 7.1: Training Example with two Epochs

when run for 2 epochs. The number does become slightly higher when we look

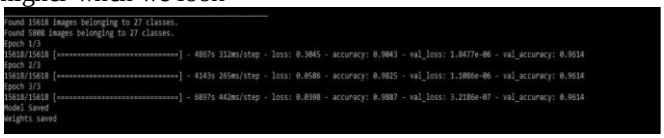

Figure 7.2: Training Example with three Epochs

at the accuracy found in case of 3 epochs i.e. 96.14%. We nd an almost similar

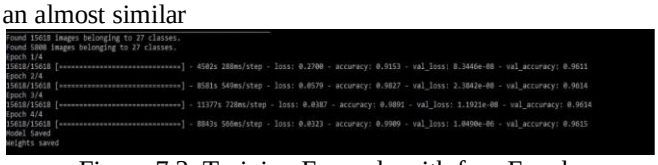

Figure 7.3: Training Example with four Epochs

result for 4 epochs. We see that the model converges on 5 epochs at 96.15%. We take this as the optimal number of iterations for the model.

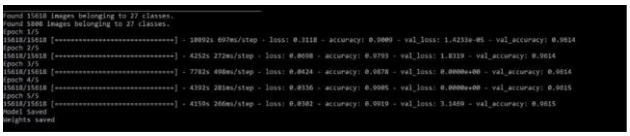

Figure 7.4: Training Example with ve Epochs

Another point to be noted is that we could have kept increasing the number of epochs expecting this constant accuracy. But on taking a closer look we see that the loss increases as we increase the epochs as well. Hence in order to keep in tone with the accuracy loss trade o we talk 5 as the optimal number of rounds for our model to train.

VIII. CONCLUSION AND FUTURE WORK

The system has been implemented with the translation from sign language to text able to recognize 36 gestures consisting of the 26 alphabets and 10 digits correctly. The text to speech module has also been implemented and is giving the correct audio output for a given input string. Few optimization techniques have been applied during the training of the model so that the system is now able to correctly identify the characters and give the correct output.

Further, we plan to integrate the sign language to text and text to speech modules to make a complete sign language to text to speech software. Also, we plan to implement a speech to text software in the same system, so that the two can work side-by-side to provide the best experience to the user. We also hope to implement an IOT solution to allow the users to use this sign language translation capability over the internet via video calling. Finally, we hope to provide users with an easy to access, cost e ective and simple platform, so that the users can communicate e ectively.

Appendix A

Appendix: Project Team Details

Project Title	**Versatile Sign Language Communicator**			
USN	Team Member's Name	Team Member's Picture	Email	Contact Number
01JST16CS015	Arunima Rashmi Giri		amazing.arunima@gmail.com	8971636350
01JST16CS093	Sanmayee Mahabaleshwar Hegde		shammy1898@gmail.com	8762152167
01JST16CS133	T S Kushal		t.s.kushal9919@gmail.com	9449938598

Figure A.1: Project Team Details

Appendix B
Appendix: COs, POs and PSOs Mapping for the ProjectWork
B.1 Course Outcomes
CO1: Formulate the problem de nition, conduct literature review and apply requirements analysis.
CO2: Develop and implement algorithms for solving the problem formulated.
CO3: Comprehend, present and defend the results of exhaustive testing and explain the major ndings.

B.2 Program Outcomes
PO1: Apply knowledge of computing, mathematics, science, and foundational engineering concepts to solve the computer engineering problems.
PO2: Identify, formulate and analyze complex engineering problems.
PO3: Plan, implement and evaluate a computer-based system to meet desired societal needs such as economic, environmental, political, healthcare and safety within realistic constraints.
PO4: Incorporate research methods to design and conduct experiments to in-vestigate real-time problems, to analyze, interpret and provide feasible conclu-sion.
PO5: Propose innovative ideas and solutions using modern tools.
PO6: Apply computing knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to professional engi-neering practice.
PO7: Analyze the local and global impact of computing on individuals and organizations for sustainable development.
PO8: Adopt ethical principles and uphold the responsibilities and norms of computer engineering practice.
PO9: Work e ectively as an individual and as a member or leader in diverse teams and in multidisciplinary domains.
PO10: E ectively communicate and comprehend.
PO11: Demonstrate and apply engineering knowledge and management prin-ciples to manage projects in multidisciplinary environments.
PO12: Recognize contemporary issues and adapt to technological changes for lifelong learning.

B.3 Program Speci c Outcomes
PSO1 Problem Solving Skills: Ability to apply standard practices and mathematical methodologies to solve computational tasks, model real world problems in the areas of database systems, system software, web technologies and Networking solutions with an appropriate knowledge of Data structures and Algorithms.

PSO2 Knowledge of Computer Systems: An understanding of the struc-ture and working of the computer systems with performance study of various computing architectures.
PSO3 Successful Career and Entrepreneurship: The ability to get ac-quaintance with the state of the art software technologies leading to entrepreneur-ship and higher studies.
PSO4 Computing and Research Ability: Ability to use knowledge in var-ious domains to identify research gaps and to provide solution to new ideas leading to innovations.

B.4 Justi cation for the mapping
The rst CO is related to problem de nition, literature survey and require-ment analysis. Planning the project such that it meets the needs of society, by considering all the constraints is very relevant for this. Investigating real time problems and incorporating our ndings in literature survey plays an important role as well. Understanding the constraints of the environment in which the system will be used plays a crucial role in deciding the requirements and using latest technology to make our implementation better is of high relevance.

The second CO, design and implementation highly depends on the way we apply already acquired knowledge about computing, mathematics, etc., the in-novation we bring into our implementation, make our implementation adaptable to technological changes that might happen in future and the way we look at the problem and apply our knowledge. The ability to analyze global and lo-cal impact of the system, ability to uphold the ethical principles of engineering practices, the way in which we communicate and comprehend the concepts, the way in which the project is handled in multidisciplinary environments hold great relevance in defending our work and explaining major ndings during our project.

SEM	SUBJECT	CODE	CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3	PSO4
VIII	Project Work	CS84P	CO1	1	3	3	1	3	2	2	2	3	1	1	1	2	3	1	1
			CO2	3	3	1	3	1	2	1	1	3	1	1	1	3	2	1	1
			CO3	2	1	3	1	1	2	3	3	3	3	2	3	1	1	3	3

Figure B.1: Assigning Priority to COs, POs and PSOs
Levels of Priority
1 - Low Relevance
2 - Medium Relevance
3 - High Relevance

The following subjects helped us in developing our project:
To formulate the problem de nition, conduct literature review and ap-ply requirements analysis that we gained the knowledge from Software Engineering and System Software.
To Develop and implement algorithms for solving the problem formulated, we gained the knowledge from subjects such as Neural Networks, Object Oriented Programming, Machine Learning, Data Warehousing and Data Mining.
To Comprehend, present and defend the results of exhaustive testing and explain the major ndings, we gained the knowledge from Engineering Mathematics, Web Technologies and Operating Systems.

BIBLIOGRAPHY

[1] Panwar.M. Hand Gesture Recognition System based on Shape parameters. In Proc. International Conference, Feb 2012.
[2] Christopher Lee and Yangsheng Xu. Online, interactive learning of gestures for human robot interfaces. Carnegie Mellon University, The Robotics Insti-tute, Pittsburgh, Pennsylvania, USA,

1996.

[3] Hyeon-Kyu Lee and Jin H. Kim. An HMM-Based Threshold Model Ap-proach for Gesture Recognition. IEEE transactions on pattern analysis and machine intelligence, Volume 21, October 1999.

[4] P. Subha Rajam and Dr. G. Balakrishnan. Real Time Indian Sign Language Recognition System to aid Deaf-dumb People. ICCT, IEEE, 2011.

[5] Olena Lomakina. Development of E ective Gesture Recognition System. TCSET'2012, Lviv-Slavske, Ukraine, February 21-24, 2012.

[6] Etsuko Ueda, Yoshio Matsumoto, Masakazu Imai, Tsulasa Ogasawara. Hand Pose Estimation for Vision based Human Interface. In Proc. 10th IEEE In-ternational Workshop on Robot and Human Communication (Roman 2001) , pp. 473-478, 2001.

[7] Claudia Nolker and Helge Ritter. Visual Recognition of Hand Postures. In Proc. International Gesture Workshop on Gesture-Based Communication in Human Computer Interaction, pp. 61-72, 1999.

[8] Meenakshi Panwar and Pawan Singh Mehra. Hand Gesture Recognition for Human Computer Interaction. In Proc. IEEE International Conference on Image Information Processing, Waknaghat, India, November 2011.

[9] Sanjay Meena. A Study of Hand Gesture Recognition Technique. Master Thesis, Department of Electronics and Communication Engineering, National Institute of Technology, India, 2011.

