

ONLINE HAND GESTURE REGOCNITION AND CLASSIFICATION FOR DEAF AND DUMB

A Project Report Phase – II

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Engineering degree in Electronics and Communication Engineering

By

A KISHORE(39130227)

KAVETI ASHOK KUMAR YADAV(39130216)



**DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
SCHOOL OF ELECTRICAL AND ELECTRONICS**

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

Accredited with Grade "A" by NAAC

**JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI - 600 119**

APRIL - 2023



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited with "A" grade by NAAC
Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119
www.sathyabama.ac.in

DEPARTMENT OF ELECTRONICS AND COMMUNICAITON ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **A KISHORE**(39130227) and **KAVETI ASHOK KUMAR YADAV** (39130216) who carried out the project entitled **"ONLINE HAND GESTURE RECOGNITION AND CLASSIFICATION FOR DEAF AND DUMB"** under our supervision from November 2022 to April 2023.

Internal Guide

Dr. K. V. KARTHIKEYAN M.E., Ph.D.,

Professor

Head of the Department

Dr. T. RAVI, M.E., Ph.D.,

Professor

Submitted for Viva voce Examination held on _____

19/04/2023

Internal Examiner

External Examiner

DECLARATION

We, **A. KISHORE** (Reg.no.39130227) and **KAVETI ASHOK KUMAR YADAV** (Reg.no.39130216) hereby declare that the Project Report entitled “**ONLINE HAND GESTURE RECOGNITION AND CLASSIFICATION FOR DEAF AND DUMB PEOPLE** ” done by under the guidance of **Dr. K. V. KARTHIKEYAN,M.E., Ph.D**, Professor, Department of Electronics and Communication Engineering at SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY, CHENNAI is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Electronics and Communication Engineering.

DATE: 19/04/2023

PLACE: CHENNAI.

1. A. Kishore.
2. K. Ashok Kumar

SIGNATURE OF THE CANDIDATE(S)

ACKNOWLEDGEMENT

We are pleased to acknowledge our sincere thanks to the **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. We are grateful to them.

We convey our thanks to **Dr. N.M. NANDHITHA, M.E., Ph.D.**, Professor & Dean, School of Electrical and Electronics and **Dr. T. RAVI, M.E., Ph.D.**, Professor & Head of Department of Electronics and Communication Engineering for providing necessary support and details at the right time during the progressive reviews.

We would like to express our sincere and deep sense of gratitude to our Project Guide **Dr. K. V. KARTHIKEYAN, M.E., Ph.D.**, Professor, Department of Electronics and Communication Engineering for his valuable guidance, suggestions and constant encouragement paved the way for the successful completion of our project work.

We wish to express our thanks to all teaching and non-teaching staff members of the Department of Electronics and Communication Engineering who were helpful in many ways for the completion of the project.

ABSTRACT

Sign Language Recognition is a breakthrough for helping deaf-mute people and has been researched for many years. Unfortunately, every research has its own limitations and is still unable to be used commercially. Some of the research has known to be successful in recognizing sign language, but requires an expensive cost to be commercialized. Nowadays, researchers have gotten more attention for developing Sign Language Recognition that can be used commercially. Researchers do their research in various ways. It starts with the data acquisition methods. The data acquisition method varies because of the cost needed for a good device, but a cheap method is needed for the Sign Language Recognition System to be commercialized.

The methods used in developing Sign Language Recognition are also varied among researchers. Each method has its own strength compared to other methods and researchers are still using different methods in developing their own Sign Language Recognition. Each method also has its own limitations compared to other methods. The aim of this paper is to review the sign language recognition approaches and find the best method that has been used by researchers. Hence other researchers can get more information about the methods used and could develop better Sign Language Application Systems in the future.

TABLE OF CONTENT

Chapter No	TITLE	Page No.
	ABSTRACT	iv
	LIST OF FIGURES	vi
1	INTRODUCTION	1
2	LITERATURE SURVEY	15
	2.1 Inferences from Literature Survey	19
	2.2 Open problems in Existing System	19
3	REQUIREMENTS ANALYSIS	20
	3.1 Feasibility Studies/Risk Analysis of the Project	20
	3.2 Software Requirements Specification Document	21
4	DESCRIPTION OF THE PROPOSED SYSTEM	22
	4.1 Selected Methodology or process model	24
	4.2 Architecture / Overall Design of Proposed System	33
	4.3 Description of Software for Implementation and Testing plan of the Proposed Model/System	34
	4.4 Project Management Plan	37
5	IMPLEMENTATION DETAILS	38
	5.1 Algorithms	38
	5.2 Testing	40
6	RESULTS	43
7	SUMMARY AND CONCLUSION	47
	7.1 Conclusion	47
	7.2 Future Enhancements	48
	7.3 Implementation	48
	REFERENCES	49

LIST OF FIGURES

FIGURE NO	FIGURE NAME	PAGE NO.
1.1	Hand shape and Hand orientation	12
4.1	Block diagram for proposed system	23
4.2	Hand Landmark	24
4.3	Alphabets hand gestures	28
4.4	Sign language hand gestures	29
4.5	System Architecture for sign language	33
6.1	Sequence Diagram	43
6.2	Data Flow Diagram	44

CHAPTER 1

INTRODUCTION

Typically, sign or signed languages are employed to communicate meaning visually. These dialects are conveyed using non-manual features in addition to physical articulations. These are completely natural languages with their own lexicon and vocabulary. Smart methods for hand motion detection from photographs in hand signals are crucial in this situation. Many people throughout the world who have speech disorders and loss of hearing can benefit from these clever techniques. For accurately recognizing symbols and signs, various approaches have indeed been documented in the literature.

Innovations like movement and face recognition have gained significant traction in the basic sign language field in recent years. Different movements known as gestures are utilized during communication. Hand or body movements are used. Gestures used in sign language typically involve visually communicated patterns. There are roughly 4,94,93,500.000 people with hearing impairments worldwide. Some of the existing methods for translating sign language take into account hand position, hand size, and finger movements.

Each sign in sign language has a particular meaning ascribed to it so that people can easily comprehend and interpret it. The people are creating distinct and unique sign languages depending on their native tongues and geographic locations. As spoken language, there are several variations in sign language. Around the world, more than 300 different sign languages are in use. They differ from one country to the next.

Sign language can have many diverse regional dialects that cause subtle variances in how people use and comprehend signs, even in nations where the same language is spoken. There are significant variances between some of the most popular sign languages, even though there are some commonalities. Additionally, not just the symptoms differ.

General Background

The use of sign language has not only been restricted to people with impaired hearing or speech to express themselves with each other or non-sign-language speakers. It has often been considered as a prominent medium of communication. Instead of acoustically conveyed sound patterns, sign language uses manual communication to convey meaning. It combines hand gestures, and facial expressions along with movements of other body parts such as eyes, legs, etc. This paper proposes a design for recognizing signs used in ASL and interpreting them. Some of the problems faced by non-speech and hard-of-hearing individuals in communication with other people were interaction-based, disparity, education, behavioural patterns, mental health, and most importantly safety concerns. The ways in which one can interact with the computer are either by using devices like a keyboard, mouse, or via audio signals, while the former always needs physical contact and the latter is prone to noise and disturbances. Physical action carried by the hand, eye or any part of the body can be considered a gesture. Hand gestures are the most convenient and interpretable way for non-speaking humans to interact.

Nonverbal communication is important in our life as it conveys about 65% of messages in comparison to verbal communication that contributes no more than 35% of our interactions. Gestures can be categorized into hand and arm gestures (recognition of hand poses, sign languages, and entertainment applications), head and face gestures (such as nodding or shaking of the head, the direction of eye gaze, opening the mouth speak, winking, and so on), and body gestures (involvement of fullbody motion).

There are numerous varieties of sign language used not only in the UK but all over the world since the speaker's nonverbal cues, actions, and visual emotions may all have a huge impact on how sign language is transmitted. Similar to spoken language, various groups, and cultures create their own means of communication specific to the environment in which they exist.

DEEP LEARNING

Deep-learning networks are distinguished from the more commonplace singlehidden layer neural networks by their depth; that is, the number of node layers through which data must pass in a multistep process of pattern recognition. Earlier versions of neural networks such as the first perceptron were shallow, composed of one input and one output layer, and at most one hidden layer in between. More than three layers (including input and output) qualify as “deep” learning. So deep is not just a buzzword to make algorithms seem like they read Sartre and listen to bands you haven’t heard of yet. It is a strictly defined term that means more than one hidden layer. In deeplearning networks, each layer of nodes trains on a distinct set of features based on the previous layer’s output. The further you advance into the neural net, the more complex the features your nodes can recognize since they aggregate and recombine features from the previous layer. This is known as feature hierarchy, and it is a hierarchy of increasing complexity and abstraction. It makes deep-learning networks capable of handling very large, high-dimensional data sets with billions of parameters that pass through nonlinear functions.

Above all, these neural nets are capable of discovering latent structures within unlabeled, unstructured data, which is the vast majority of data in the world. Another word for unstructured data is raw media; i.e., pictures, texts, video, and audio recordings. Therefore, one of the problems deep learning solves best is in processing and clustering the world’s raw, unlabeled media, discerning similarities and anomalies in data that no human has organized in a relational database or ever put a name to. For example, deep learning can take a million images, and cluster them according to their similarities: cats in one corner, ice breakers in another, and in a third all the photos of your grandmother. This is the basis of so-called smart photo albums. Deep-learning networks perform automatic feature extraction without human intervention, unlike most traditional machine-learning algorithms.

Given that feature, extraction is a task that can take teams of data scientists years to accomplish, deep learning is a way to circumvent the chokepoint of limited experts. It augments the powers of small data science teams, which by their nature do not scale. When training on unlabeled data, each node layer in a deep network

learns features automatically by repeatedly trying to reconstruct the input from which it draws its samples, attempting to minimize the difference between the network's guesses and the probability distribution of the input data itself. Restricted Boltzmann machines, for example, create so-called reconstructions in this manner. In the process, these neural networks learn to recognize correlations between certain relevant features and optimal results – they draw connections between feature signals and what those features represent, whether it be a full reconstruction, or with labeled data. A deep-learning network trained on labeled data can then be applied to unstructured data, giving it access to machine-learning.

NEURAL NETWORKS

A neural network is a series of algorithms that endeavors to recognize underlying relationships in a set of data through a process that mimics the way the human brain operates. In this sense, neural networks refer to systems of neurons, either organic or artificial in nature. Neural networks can adapt to changing input; so the network generates the best possible result without needing to redesign the output criteria. The concept of neural networks, which has its roots in artificial intelligence, is swiftly gaining popularity in the development of trading systems. A neural network works similarly to the human brain's neural network. A "neuron" in a neural network is a mathematical function that collects and classifies information according to a specific architecture. The network bears a strong resemblance to statistical methods such as curve fitting and regression analysis. A neural network contains layers of interconnected nodes. Each node is a perceptron and is similar to multiple linear regression. The perceptron feeds the signal produced by a multiple linear regression into an activation function that may be nonlinear. ¹⁴ In a multi-layered perceptron (MLP), perceptrons are arranged in interconnected layers. The input layer collects input patterns. The output layer has classifications or output signals to which input patterns may map. Hidden layers fine-tune the input weightings until the neural network's margin of error is minimal. It is hypothesized that hidden layers extrapolate salient features in the input data that have predictive power regarding the outputs.

CNN

Convolutional Neural Network is an algorithm of Deep Learning. That is used for Image Recognition and in Natural Language Processing. Convolutional Neural Network (CNN) takes an image to identify its features and predict it. It captures the spatial and temporal dependencies of the image. Each CNN layer learns filters of increasing complexity. The first layers learn basic feature detection filters like edges, corners. The middle layer learns filters that detect parts of objects. The last layers have higher representations they learn to recognize full objects in different shapes and positions. Suppose, when you see some image of a Dog, your brain focuses on certain features of the dog to identify. These features may be the dog's ears, eyes, or it may be anything else. Based on these features your brain gives you a signal that this is a dog. Similarly, Convolutional Neural Network processes the image and identifies it based on certain features. Convolutional Neural Network is gaining so much popularity over artificial neural networks.

Steps In a Convolutional Neural Network,

1.Convolution Operation.

2.RELU Layer

3.Pooling

4.Flattening

5.Full Connection

Generally , a convolution neural network architecture generally has several components:

- A convolution layer - you can think of this layer as “what relevant features are we picking up in an image?” In a convolutional neural network, we have multiple convolutional layers that extract low to high-level features depending on what

specific layer we are focusing on. To give an (over-simplified) intuition, earlier convolutional layers pick up lower-level features (i.e. like lines and edges) while later convolutional layers pick up higher-level features based on inputs from lower-level features (i.e. shapes, structures) - analogous to how vision works in the human brain.

- A pooling layer - convolutional neural networks are typically used for image classification. However, images are high-dimensional data - so we would prefer to reduce the dimensionality to minimize the possibility of overfitting. Pooling essentially reduces the spatial dimensions of the image based on certain mathematical operations such as average or max-pooling (there's a nice graphic here). We generally incorporate pooling since it

(1) generally, acts as a noise suppressant

(2) makes it invariant to translation movement for image classification and

(3) helps capture essential structural features of the represented images without being bogged down by the fine details.

- Fully-connected layer - You can think of a series of convolution and pooling operations as dimensionality reduction steps prior to passing this information over to the fully connected (Dense) layer. Essentially, what the fully connected layer does is that it takes the "compressed" representation of the image and it tries to fit a basic NN (multi-layer perceptron) when doing classification.

The change in paradigm that the usage of NNs encompasses is a very important one. Prior to the usage of NNs in image classification, the practitioner had to use explicitly coded algorithms for detecting features. This was a job that often involved a lot of work. Machine Learning was not used throughout the whole system, only the last layer of the system, the classifier, had the capacity to learn and adapt itself to the specific problem. NNs, and especially CNNs, dramatically change that. CNN's extract features from images and learns how to do it. The practitioner doesn't need to craft complicated hard-coded algorithms to extract those features. Since the introduction

of the first CNNs in 2012, all the winning teams for the classification task have used different types of CNNs. CNN's systems now clearly outperform other image classification systems such as the ones described. State-of-the-art CNNs have even crossed the boundary of what is considered to be the error rate for human beings in image classification tasks. Ever since its introduction in 2012, CNNs have been making steady progress in decreasing the error rate in both tasks. This is due to the introduction of new strategies that further improve the convergence and training speed of CNNs. CNNs are a particular kind of NNs that give their name to the fact that they make extensive use of Convolutional Layers.

Convolutional Layers consist of a learnable filter of fixed size (kernel) to be applied to images. The name comes from the fact that at each forward pass, the learned filter is convolved with the image resulting on a 2-dimensional map. Another type of layer widely used in CNNs are the pooling layers. These layers perform a type of down sampling on the input signal. There are various types of Pooling layers, max-pooling being the most common. ReLUs or Rectified Linear Units, are also widely used in CNNs. ReLUs are units that apply a function similar to Equation 2.1. There are often alternatives for these types of units, Equation 2.2 and 2.3 show two of them.

$$\text{ReLU} : f(x) = \max(0, x) \quad (2.1)$$

$$\text{Tanh} : f(x) = \tanh(x) \quad (2.2)$$

$$\text{Sigm} : f(x) = (1 + e^{-x})^{-1} \quad (2.3)$$

Fully Connected Layers are used too, as in more standard NNs. These layers consist of an array of neurons, each of which is connected to every single output of the previous layer. Loss Layers are typically used at the end of the CNN to figure out the penalty for a given output and to provide feedback used by the CNN to learn. In addition to these more standard set of layers, modern CNNs make use of a very extensive set of techniques and strategies that greatly enhance their performance. Dropout is one of such techniques. It consists on temporarily deactivating some neurons within the net, preventing it from over-fitting. This temporary deactivation of neurons is typically applied only to 8 Fully-connected layers. CNNs nowadays also make extensive use of GPU processing power that allows for a faster training, an idea pioneered in 2012 that is now a standard. Evidence shows that one way of

improving a CNN performance is by stacking more layers and making the network effectively deeper the problem that often arises is that as more layers are stacked, the performance saturates and eventually starts to decrease at some point. To address this issue, the idea of deep residual networks has been proposed. Residual Networks try, with minor changes to the architecture, and with virtually no impact on performance, to approximate residual functions instead of standard functions.

If we have a CNN with input x and we want the output to approximate a function $O(x)$, we can instead, approximate $F(x) = O(x) - x$ and then compute $O(x)$ from $F(x)$ since $O(x) = F(x) + x$. Evidence shows that CNNs train better and converge faster when the approximated function is in fact $F(x)$, a residual function. This approach opens door for usage of even deeper and more powerful models. Some pre-processing techniques are also used to improve the data fed to the CNN. These techniques include normalization and zero-centering of the data. These techniques are mainly used in order to overcome the vanishing and exploding gradient problem, that often affects NN systems' training. Both problems arise when models are trained using gradientbased methods. Gradients are often computed using the chain rule. In case of nearzero gradients, this lead to several very small numbers being multiplied together in order to calculate the changes in the last layers. This causes the last layers to suffer virtually no change at all in its outputs as the training progresses. The opposite happens when activation functions have high derivative values, leading 9 Related Work to the uncontrolled increase of the gradient of the last layers. As NN systems tend to require a lot of examples, better results are obtained when data augmentation techniques are used. CNN frameworks often incorporate mechanisms to augment data. Cropping is a very simple technique where several crops of a single image are fed to the NN, augmenting the dataset by a factor equal to the number of crops. Rotations [KSH12] of the image are also used, augmenting the dataset by a factor equal to the number of rotations done.

Objective

Like all other time-variable signs and signals, it is not easy to compare gestures directly in the Euclidean space. This is because of the time dependencies and a large no. of irrelevant areas in the frames. it is very difficult to find real representative hand engineered features for hand gestures. To work well with conventional

classifiers, the required features should involve robust descriptors for hand shape, position, orientation, and temporal dependence between consecutive frames. These features should also be robust against different circumstances such as occlusion and background clutter.

For these reasons, deep learning is employed in this article, since it seems to be a promising solution. The CNN had been successfully utilized for automatic feature learning. It achieved excellent results in image classification, object recognition, and even in human activity recognition.

The main contributions in this article, are as follows:

1. Presenting a CNN model for learning Spatio-temporal features for hand gestures.
2. Adapting a pretrained deep model to work well on hand gesture recognition.

For instance, native English speakers can be found in both Britain and the United States. However, there are considerable differences between American and British sign languages. Here is where a lot of companies and institutions still have trouble reaching out to the Deaf and Hard of Hearing people.

Here are a few different types of sign language as mentioned below:

Indian Sign Language

Indian Sign Language uses both the right and left hands to depict a variety of hand gestures. Indian Sign Language serves as both a communication tool for the deaf and a representation of their pride and identity. The Rights of Persons with Disabilities (RPwD) Act 2016, which took effect on April 19, 2017, acknowledges sign language as a form of interaction that is particularly helpful when speaking with people who have hearing loss.

Authorities are also required under the Act to encourage the use of sign language so that people with hearing loss can take the role in society and benefit from it. The ISL gradually changed from 4000 words to 10,000 words and changed its goal of being used more widely.

American Sign Language (ASL)

On the other hand, one hand is all that's required to sign in ASL. As a result, putting the system into place is made simple. ASL has its own growth path and is independent of all spoken languages. The main sign language used by Deaf people in the United States and most of Anglophone Canada is American Sign Language (ASL), a natural language.

ASL is a comprehensive and well-structured visual language that uses both manual and non-manual characteristics to communicate. Dialects of ASL and creoles based on ASL are spoken around the world, including much of West Africa and portions of Southeast Asia, besides North America.

As a lingua franca, ASL is also commonly studied as a second or foreign language. French Sign Language is most closely similar to ASL. Although ASL exhibits characteristics untypical of creole languages, like agglutinative morphology, it has been argued that ASL is a creole language of the FSL.

French Sign Language (FSL)

The gesture used by the deaf in France and the French-speaking regions of Switzerland is known as French Sign Language (LSF). Ethnology estimate that 100,000 native signers are involved.

Dutch Sign Language (NGT), Flemish Sign Language (VGT), Belgian-French Sign Language (LSFB), Irish Sign Language (ISL), American Sign Language (ASL), Quebec (also known as French Canadian) Sign Language (LSQ), Brazilian Sign Language (LSB, LGB or LSCB), and Russian Sign Language are all linked to and partly descended from French Sign Language (RSL).

The invention of French Sign Language is usually, though incorrectly, credited to Charles Michel de l'Épée (abbé de l'Épée). A pair of deaf twin sisters were inside a nearby house when he entered, seeking shelter from the rain. He was immediately struck by the depth and sophistication of the language they were using to communicate with one another and the deaf society in Paris. In fact, he is said to have discovered the language by complete accident.

The abbé began studying what is now known as Old French Sign Language, and eventually, he founded a free education for the hearing impaired. He created a methodology at this school to instruct his children how to read and write that he dubbed "methodical symbols."

British Sign Language (BSL)

The Deaf community in the *UK* uses the sign language known as British Sign Language (BSL), which is their first or chosen language. The British Deaf Association calculates that there are 151,000 BSL users in the UK, of whom 87,000 are Deaf, based on the percentage of people who reported "using British Sign Language at home" on the 2011 Scottish Census.

In comparison, 15,000 persons who resided in England and Wales identified using BSL as their primary language in the 2011 England and Wales Census. As having to hear relatives of the deaf, basic sign translators, or as a consequence of other interactions with the British Deaf community, individuals who aren't deaf might also use BSL.

Elements Of Sign Language

SL is a visual-spatial language based on positional and visual cues, including arm and body movements, the position and orientation of the hands, and the shape of the fingers and hands. Together, these elements are employed to communicate an idea's meaning. There are typically five components to the phonological structure of SL. In Second Life, each motion is made up of five different components. These five building pieces serve as SL's most valuable components, and automated, intelligent systems can use them to recognize SL.

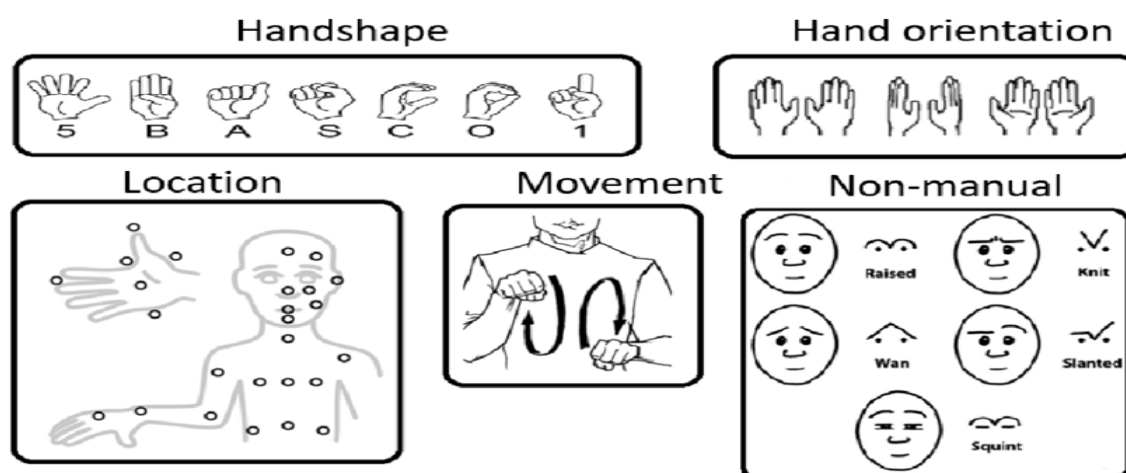


Fig 1.1 Hand shape and Hand Orientation

Methods Of Sign Language Detection

Scholarly approaches to overcoming challenges brought on by disabilities are numerous, methodical, and context-specific. SLR systems, which are used to translate the signals of SL into text or speech to establish communication with individuals who do not understand these signs, are one of the crucial interventions. One of the most important projects aimed at collecting data for the motion of human hands is the development of SLR systems based on the sensory glove. To record hand layouts and identify the related meanings of gestures, three strategies are used: vision-based, sensor-based, and a mix of the two.

Vision-Based Strategies

Lenses are the main equipment used by vision-based systems to gather the required input data. The primary benefit of employing a camera is that it eliminates the requirement for sensors in sensory gloves and lowers the system's construction expenses. Due to the blur produced by a web camera, most laptops employ a high-end camera, which is quite inexpensive.

Despite the high-quality cameras that are found in most smartphones, there are still a number of issues that hinder the creation of real-time recognition applications. These issues include the limited field of view of the capturing device, high computational costs, and the requirement for multiple cameras in order to capture robust results (due to an issue of complexity and obstruction).

Sensory-Based Strategies

An alternate method for gathering information on gestures is to utilize a certain kind of instrumented glove that is equipped with flexion (or bend) sensors, accelerometers (ACCs), proximity sensors, and abduction sensors.

The bend angles of the fingers, the abduction between the fingers, and the wrist's orientation (roll, pitch, and yaw) are all measured using these sensors. Depending on the number of sensors incorporated into the glove, different degrees of freedom (DoF) can be achieved using such gloves, ranging from 5 to 22.

Glove-based systems have a significant benefit oversight systems in that they may immediately report necessary and pertinent information (degree of bend, pitch, etc.) to the portable computer in the form of voltage values obviating the need to transform raw data into useful values.

Contrarily, the computational overhead of vision-based solutions is increased by the requirement to apply particular tracking and feature extraction techniques to raw streaming video.

Hybrid-Based Strategies

The third way integrates camera- and glove-based technologies in a hybrid design to gather raw gesture data. To improve the accuracy and precision overall, this method employs mutual mistake minimization.

Unfortunately, due to the expense and computational cost of the complete apparatus, little research has been done in this area. Furthermore, when combined with hybrid tracking techniques, immersive virtual reality solutions show promise.

CHAPTER 2

LITERATURE SURVEY

Soma Shrenika, Myneni Madhu Bala, “Sign Language Recognition Using Template Matching Technique”, International Conference on Computer Science, Engineering and Applications (ICCSEA), 2020

Normal people cannot understand the signs used by the deaf because they do not understand the meaning of each sign. This is the goal of the system proposed here. This system employs a camera to record various hand gestures. Following that, the image is processed using various algorithms. Initially, the image is pre-processed. The edges are then determined using an edge detection algorithm. Finally, the sign is identified and the text is displayed using a template-matching algorithm. Because the output is text, the meaning of a specific sign can be easily interpreted. The system is implemented in Python using OpenCV.

H Muthu Mariappan, V Gomathi, “Real-Time Recognition of Indian Sign Language”, International Conference on Computational Intelligence in Data Science (ICCIDS), 2019

The skin segmentation feature of OpenCV is used to identify and track the Regions of Interest (ROI) for recognizing the signs. The fuzzy c-means clustering machine learning algorithm is used to train and predict hand gestures. Many applications for gesture recognition exist, including gesture-controlled robots and automated homes, game control, Human-Computer Interaction (HCI), and sign language interpretation. The proposed system can recognize real-time signs. As a result, it is extremely beneficial for hearing and speech impaired people to communicate with normal people.

Wanbo Li, Hang Pu, Ruijuan Wang, “Sign Language Recognition Based on Computer Vision”, IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA), 2021

This system makes use of a PyQt-designed GUI interface. Users can choose between sign language recognition and translation, image capture with OpenCV, and special processing with the trained CNN neural network. Through LSTM decisions, the model can then identify American sign language. The user can also use the voice button to have the system convert the corresponding gesture image into the same pixels and write it to the video file based on the user's voice. The experimental results show that the rate of sign language recognition is 95.52% when compared to similar algorithms [5, 6], and the rate of sign language [7] (American sign language and Arabic numerals) is 90.3%

Pushkar Kurhekar, Janvi Phadtare, Sourav Sinha, Kavita P. Shirsat, "Real Time Sign Language Estimation System", 3rd International Conference on Trends in Electronics and Informatics (ICOEI), 2019

This paper describes a system that allows people with speech and hearing impairments to communicate freely. The model can extract signs from videos by processing them frame by frame in a minimally cluttered environment. This sign is then displayed in text form. For webcam input and displaying the predicted sign, the system employs a Convolutional Neural Network (CNN) and fastai - a deep learning library - in conjunction with OpenCV. The experimental results show good sign segmentation in various backgrounds and high accuracy in gesture recognition

Aniket Kumar, Mehul Madaan, Shubham Kumar, Aniket Saha, Suman Yadav, "Indian Sign Language Gesture Recognition in Real-Time using Convolutional Neural Networks", 8th International Conference on Signal Processing and Integrated Networks (SPIN), 2021

Using Convolutional Neural Networks, this paper proposes a model for identifying and classifying Indian Sign Language gestures in real-time (CNN). The model, which was built with OpenCV and Keras CNN implementations, aims to classify 36 ISL gestures representing 0-9 numbers and A-Z alphabets by transforming them to text equivalents. The dataset that was created and used consists of 300 images for each

gesture that were fed into the CNN model for training and testing. The proposed model was implemented successfully and achieved 99.91% accuracy for the test images.

Sai Nikhilesh Reddy Karna, Jai Surya Kode, Suneel Nadipalli, Sudha Yadav, “American Sign Language Static Gesture Recognition using Deep Learning and Computer Vision”, 2nd International Conference on Smart Electronics and Communication (ICOSEC), 2021

This study proposes a real-time hand-gesture recognition system based on the American Sign Language (ASL) dataset, with data captured using a BGR webcam and processed using Computer Vision (OpenCV). The Vision Transformer Model was used to train the 29 static gestures (the alphabet) from the official, standard ASL dataset (ViT). After getting trained with 87,000 RGB samples for 1 epoch, the model had an accuracy rate of 99.99%. (2719 batches of 32 images each).

Shirin Sultana Shanta, Saif Taifur Anwar, Md. Rayhanul Kabir, “Bangla Sign Language Detection Using SIFT and CNN”, 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT), 2018

There have been very few studies on detecting Bangla sign language. The classifiers used by the majority of researchers in this field were SVM, ANN, or KNN. In this paper, the researchers attempt to develop a Bangla sign language system that uses SIFT feature extraction and Convolutional Neural Network (CNN) classification. The researchers also show that using the SIFT feature improves CNN's detection of Bangla Sign language.

Rajiv Ranjan, B Shivalal Patro, Mohammad Daim Khan, Manas Chandan Behera, Raushan Kumar, Utsav Raj, “A Review on Sign Language Recognition Systems”, IEEE 2nd International Conference on Applied Electromagnetics, Signal Processing, & Communication (AESPC), 2022

For the enhancement of the recognized gesture, open CV, Gaussian blur, and contour approximation techniques are used. The TensorFlow framework is best suited for accurate software prediction with a CNN model of VGG-16 architecture and the Transfer Learning training method. The use of these tools results in a more accurate prediction of sign language gestures in the English alphabet. Given the project's future scope, the technology stack is both robust and scalable. This review paper discusses various methods and structures for sign language recognition.

Kohsheen Tiku, Jayshree Maloo, Aishwarya Ramesh, Indra R, “Real-time Conversion of Sign Language to Text and Speech”, Second International Conference on Inventive Research in Computing Applications (ICIRCA), 2020

The purpose of this paper is to examine the performance of various techniques for converting sign language to text/speech format. Following analysis, an Android application is created that can convert real-time ASL (American Sign Language) signs to text/speech.

Vishwa Hariharan Iyer, U.M Prakash, Aashrut Vijay, P. Sathishkumar, “Sign Language Detection using Action Recognition”, 2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE), 2022

This proposal is centered on the continuous detection of image frames in real-time using action detection to detect the user's action. After recognising key points using mediapipe holistic, which includes face, pose, and hand features, the model employs an LSTM neural network model. Collecting key value points for training and testing, pre-processing the data, and creating labels and features are all part of the proposed work. It saves the weights and uses confusion matrix accuracy to evaluate the model.

2.1 INFERENCES FROM LITREATURE SURVEY

Related works that was done have limitations as well as scope to be developed. The main limitations of these paper, there was no combined guidelines to help a visually impaired people and deaf people to communicate in sign language effectively. Our proposed system is dealt with to solve the limitations of these papers and providing a proper guideline also always forewarn from the obstacle faced simultaneously.

2.2 OPEN PROBLEMS IN EXISTING SYSTEM

In face of the problems often found with other learning models, such as the presence of multiple local minima and the curse of dimensionality, existing methods have been significantly ineffective therefore. The SVM method does not have any edge over the curse of dimensionality, thus their practical importance can be diminished. An immediate problem arising from the SVM's original hyperplane formulation is that it is not very obvious how to make the model applicable to more than two classes. Several approaches have been proposed to overcome this limitation, two of them being known as the 1-vs-1 and 1-vs-all strategies for multiple class classification. For a decision problem over classes, 1-vs-all requires the creation of classifiers, each trained to distinguish one class from the others. The decision then is taken in a winner-takes-all approach. However, there is no clear indication this approach results in an optimum decision. As a result, we have proposed a model that solves these problems and offers higher accuracy and reliability as compared to traditional classifiers method.

CHAPTER 3

REQUIREMENT ANALYSIS

3.1 FEASIBILITY STUDIES/RISK ANALYSIS OF THE PROJECT

3.1.1 FEASIBILITY STUDY

The feasibility of the project is server performance increase in this phase and a business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis, the feasibility study of the proposed system is to be carried out. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- Economic feasibility
- Technical feasibility
- Operational feasibility

ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of funds that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands being placed on the client. The developed system must have modest requirements, as only minimal or null changes are required for implementing this system.

OPERATIONAL FEASIBILITY

The aspect of the study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

3.2 SOFTWARE REQUIREMENTS SPECIFICATION DOCUMENT

S/W CONFIGURATION:

- Operating System : Windows/Linux, Mac
- Programming Language : python 3.6
- IDE : Anaconda / Jupiter notebook
- Packages/libraries : Tensor flow, NumPy, OpenCV

H/W CONFIGURATION:

- Processor : I3/Intel Processor
- Hard Disk : 160GB
- RAM : 8GB

CHAPTER 4

DESCRIPTION OF PROPOSED SYSTEM

In the proposed model, the objective is to recognize all 26 Alphabets with the aid of hand gestures through a web camera. We will create an Alphabet detection Model using Tensor flow Dense model to detect angled and normal alphabets to train our dataset that stands for alphabets. This deep learning model will recognize the alphabet with gestures captured in real time via a webcam. We will create a gesture detection Model using Tensor flow Dense model to detect sign language hand gestures to train our dataset that stands for hand gestures. This deep learning model will recognize the sign language with hand gestures captured in real time via a webcam. For this system, we will create a Node JS webapp to take webcam input using OpenCV and detect alphabets and gestures using our trained model.

A very well-known technique which has worked effectively in case of small labelled data is transfer learning. A network which is trained on a source task is used as a feature extractor for target task. There are many CNN models trained on ImageNet which are available publicly such as VGG-16, VGG-19, Inception, Resnet. Transferable feature representation learned by CNN minimizes the effect of over-fitting in case of small, labelled set.

For the implementation of this project 4 algorithms were considered, VGG16, VGG19, Resnet50, Inception V3. VGG16 being effective in terms of its object detection(car) capability and classification (severity and location) because of its simple linear architecture and hence compatibility with the required use case. Hence VGG16 model was best suitable for implementation.

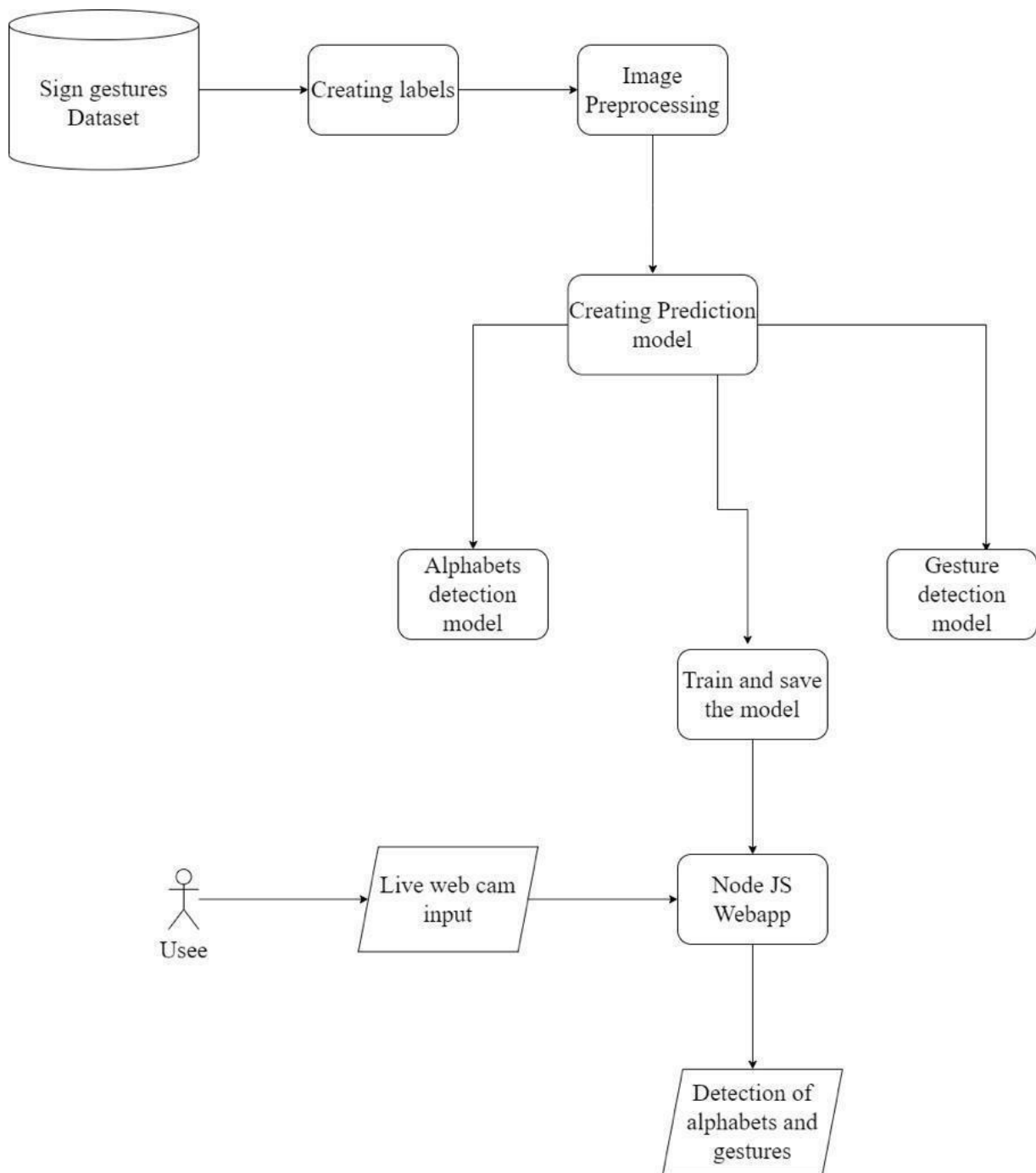


FIGURE 4.1 BLOCK DAIGRAM FOR PROPOSED SYSTEM

4.1 SELECTED METHODOLOGY OR PROCESS MODEL

4.1.1 Module 1: Data Gathering and Data Preprocessing

Creating Hand Landmark Creator

We will create a landmark creator function to detect the left and right hand from the frame and create a landmark for the hand. The location of hand landmarks is an important source of information to recognize hand gestures, effectively exploited in a number of recent methods that work from depth maps. A hand landmark model processes a cropped bounding box image and returns three-dimensional hand key points on hand.

This function reaches the exact location of 21 key points with a 3D hand knuckle coordinate which is driven inside the hand regions detected by regression which will produce the prediction of direct coordinates which is a model of the hand landmark.



Figure 4.2: Hand landmark

Each hand-knuckle of the landmark coordinate is composed of x, y, and z where x and y are normalized to [0.0, 1.0] by image width and height, while z represents the depth of the landmark. The depth of the landmark that can be found on the wrist being the ancestor. The closer the landmark is to the camera; the value becomes smaller.

Data Gathering

The process of collecting data is crucial for the development of an effective ML model. The quality and quantity of your dataset has a direct effect on the decision-making process of the AI model. And these two factors influence the robustness, precision and performance of AI algorithms. As a consequence, data collection and structuring often takes longer than data model training. We will gather data using webcam only for signlanguage that will contain alphabets (A-Z) and different gestures, i.e. Hello, Thank you, Eat, Goodbye and many more.

Creating Labels for Detections

Image labeling is a type of data labeling that places emphasis on the identification and labeling of specific details in an image. In computer vision, data labeling involves adding tags to raw data like pictures and videos. Each tag stands for an object class associated with the data. Supervised ML models use labels during learning to identify a specific object class in unclassified data. It helps these models combine meaning with data, which can help form a model. Image annotation is used to create data sets for computer vision models that are divided into training sets, originally used to train the model and test and validation sets used to assess model performance. Data scientists use the dataset to shape and evaluate their model, then the model can automatically label unlabeled and unseen data.

We will create labels for each image to classify the alphabets and gestures. Image labeling is an essential function of computer vision algorithms. The ultimate goal of machine learning algorithms is to obtain automatic labeling, but to train a model will require a large set of pre-labeled image data.

METHODS FOR IMAGE LABELING :

Manual image annotations

This is the process of manually defining labels for a complete image or drawing regions in an image and adding text descriptions for each region. Manual annotation is typically supported by tools that enable operators to rotate a large number of images, draw regions on an image, and label and store these data in a standard .

Automated image annotations

Automated annotation tools can help manual annotators by trying to find object limits in an image and providing a starting point for annotation. Automated annotation algorithms are not entirely precise, but they can save time for human annotators by providing at least one partial map of the objects in the image.

Synthetic image labeling

It is a precise, cost-effective technique that can replace manual annotations. This involves automatically generating images similar to actual data, in accordance with the criteria defined by the operator. There are three common methods of producing synthetic images:

Variational Autoencoders (VAE): these are algorithms that start from existing data, create a new data distribution, and link it to the original space using an encoder-decoder method.

Generative Adversarial Networks (GAN): these are models between two neural networks. One neural network tries to create fake images, while the other tries to distinguish actual images from fake ones. Over time, the system can produce photorealistic images which are difficult to distinguish from real images.

Neural Radiance Fields (NeRF): This model takes a series of images describing a 3D scene and automatically makes additional new perspectives from the same scene. It works by calculating a 5-dimensional radiation function to generate each voxel on the target image.

Data Preprocessing

Algorithms which learn data are just statistical equations operating on database values. Thus, as the saying goes, "if the garbage enters, the garbage leaves". Your data project can only achieve success if the data entering the machines is of high quality.

Data from actual scenarios always contain noise and missing values. This occurs due to manual mistakes, unexpected events, technical problems, or various other obstacles. Incomplete and noisy data cannot be consumed by algorithms, as they are generally not designed to handle missing values, and noise causes disruption to the actual sample setup.

Data preprocessing is a step in transforming raw data so that problems due to incompleteness, inconsistency, and/or lack of proper representation of trends are resolved in order to obtain a data set in a comprehensible format. In our model, for data preprocessing we will convert our labels to categorical values using a One Hot Encoding method. As we will be taking raw webcam frames, we won't do any other preprocessing method.

Categorical Encoding

Sometimes the data is in a form that cannot be processed by machines. For example, a column with string values, like names, will not mean anything for a model that depends solely on numbers. We, therefore, have to process the data to assist the model in interpreting them. It is referred to as categorical encoding. There are multiple ways in which we can encode categories. Here we use the One Hot Encoding method.

When data is not in an inherent order, we can use only one hot encoding. A hot encoding generates a column for each category and assigns a positive value, 1, in any row within that category, and 0 when not present. The downside is that multiple features are generated from a single feature, which makes the data large. It doesn't matter if we don't have too many features.

4.1.2 Module 2: Model Creation and Training

Alphabet Detection Model

The objective is to recognize all 26 Alphabets with the aid of hand gestures through a web camera. We will create an Alphabet detection Model using the Tensorflow Dense

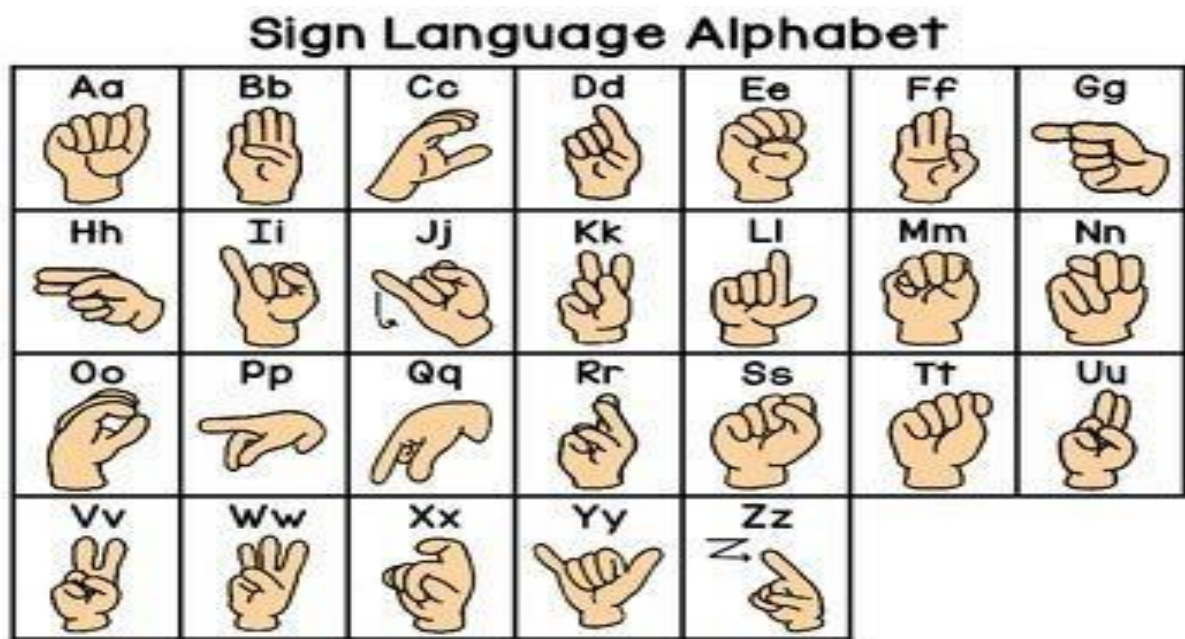


Figure 4.3: Alphabet hand gestures

The dense layer is a neural network with a deep connection, which means that each neuron in the dense layer receives the input of all neurons from its previous layer. Dense Layer does a matrix-vector multiplication, and the values used in the matrix are parameters that can be trained and updated with the help of backpropagation. The output produced by the dense layer is a vector of dimension 'n'. A dense Layer is used to change the dimensions, rotate, scale, and translate the vector.

Gesture Detection Model

We will create a gesture detection Model using the Tensorflow Dense model to detect sign language hand gestures to train our dataset that stands for hand gestures. This deep learning model will recognize the sign language with hand gestures captured in real-time via a webcam.



Figure 4.4: Sign language hand gestures

Tensor flow dense is the kind of layer and function available in neural networks whilst implementing artificial intelligence and deep learning in a python programming language. There are deep connections that exist between the neurons in the neural network in dense layers. The pattern they follow is such that each individual neuron gets data input from all the neurons of the previous layer, forming the complex model. While on the other end, dense is also a function used in the neural networks of TensorFlow, which produces the output by applying activation of the dot of Kernel and input and adding the bias effect to it. At the same time, dense is also a function used in TensorFlow neural networks which produce the output by applying kernel point activation and input and adding a bias effect.

Data Splitting

Data splitting is the time at which data is divided into two or more subsets. Typically, with a two-part split, one part is used to evaluate or test the data and another to form the model. Data splitting is an important part of data science,

particularly for the creation of data-based models. This technique ensures that data and process models that utilize data models, such as machine learning, are accurately created.

In a basic two-part split, the training dataset is used for training and model development. Training sets are routinely used to estimate different parameters or to compare the performance of different models. The test data set is used at the end of the training. Training and test data are compared to make sure the final model is working properly. When using machine learning, data is usually divided into three or more sets. In three sets, the extra set is the development set, which is used to modify the parameters of the learning process. Organizations and data modelers may elect to separate split data using data sampling methods, such as the following three methods:

Random sampling: This data sampling method protects the data modeling process against bias to different data characteristics that may occur. However, random splitting can pose problems in terms of uneven distribution of data.

Stratified random sampling: It selects random data samples from specific parameters. Ensures data is properly distributed across training and test sets.

Nonrandom sampling: This approach is generally used when data modelers are looking for the latest data as a test set.

With data splitting, organizations do not have to choose between using data for statistical analysis and analytics, as the same data can be used in the various processes.

Training Model

Model training is the lifecycle phase of data science development during which practitioners attempt to figure out the best combination of weight and bias to a machine learning algorithm to minimize a loss function on the prediction range. The goal of model development is to construct the best mathematical representation of the relationship between data features and a target label (in supervised learning) or among the features themselves (in unsupervised learning).

Model training is the key step of machine learning leading to a model ready for validation, testing, and deployment. The performance of the model determines the quality of the applications which are constructed using it. The quality of the training data and the training algorithm are significant assets in the training phase of the model. Training information is usually split for training, validation, and testing. The training algorithm is selected as a function of the end-use case. There are a number of trade-off points for choosing the best algorithm — model complexity, interpretability, performance, computational requirements, etc. All these aspects of model training make it a process that is both involved and important for the overall development cycle of machine learning.

Saving the Best Model

Saving our machine-learning models is an important step in the machine-learning workflow, so we can reuse them in the future. For example, it is very likely that we will have to compare models to determine which champion model to use in production. Saving the models while they are being trained facilitates this process. The alternative would be to form the model whenever it is to be used, which can greatly affect productivity, especially if the model takes a long time to form.

4.1.3 Module 3: Creating Node JS Web App

An important part of building a machine learning model is to share the model we have built with others. No matter how many models we create, if they remain offline, very few people will be able to see what we are achieving. That is why we should deploy our models, so that anyone can play with them through a nice User Interface (UI). For this system, we will create a Node JS webapp to take webcam input using OpenCV and detect alphabets and gestures using our trained model.

Nodejs is a server-side platform powered by Google Chrome JavaScript Engine (V8 Engine). Node.js is a platform built on Chrome JavaScript runtime to build fast, scalable network applications with ease. Node.js uses a non-blocking, event-driven I/O model that makes it lightweight and efficient, perfect for real-time, data-intensive applications that operate on distributed devices. It is an open-source, cross-platform runtime environment for developing server-side and networking applications.

on OS X, Microsoft Windows, and Linux. Node.js also provides a rich library of different JavaScript modules that simplifies the development of web applications using Node.js to a great degree.

Node.js = Runtime Environment + JavaScript Library

These are just a few of the key features that make Node.js the first choice for software architects:

Asynchronous and Event Driven: All APIs of the Node.js library are asynchronous, meaning non-blocking. This basically means that a Node.js- based server never waits until an API returns data. The server moves to the next API after calling it and a Node.js event notification mechanism assists the server to get a response from the previous API call.

Very Fast: Being constructed on Google Chrome V8 JavaScript Engine, the Node.js library is very fast in code execution.

Single-Threaded but Highly Scalable: Node.js has a single-threaded

model with an event loop. The event mechanism helps the server to respond in a nonblocking manner and makes the server highly scalable as opposed to traditional servers that create limited threads to manage requests. Node.js uses a single-threaded program and the same program can serve a lot more requests than traditional servers like Apache HTTP Server.

No Buffering: No Node.js application buffers the data. These applications simply generate data in chunks.

License: Node.js is licensed by MIT

4.2 ARCHITECTURE / OVERALL DESIGN OF THE PROPOSED SYSTEM

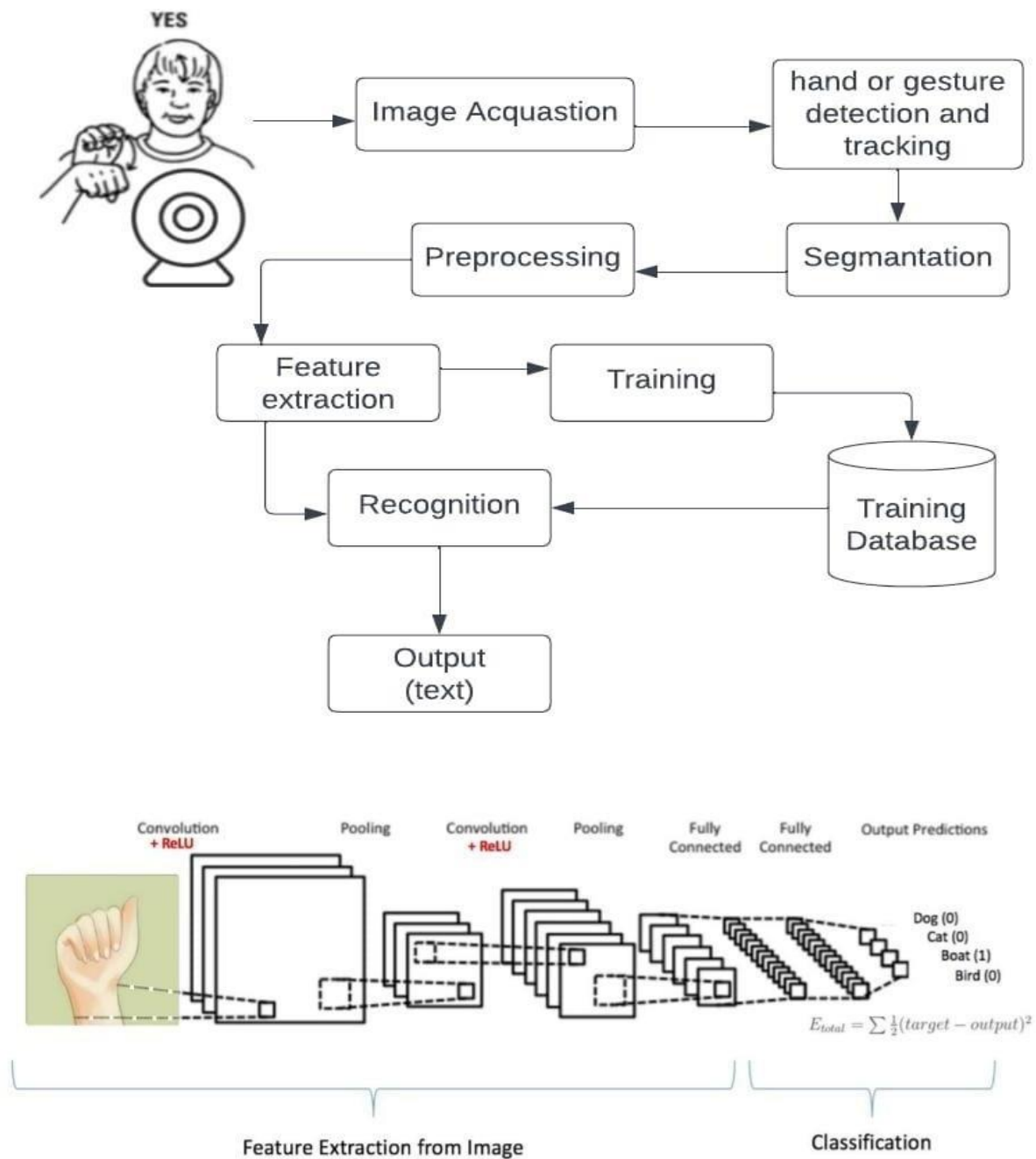


Fig 4.5: System Architecture for Sign Language

4.3 DESCRIPTION OF SOFTWARE FOR IMPLEMENTATION AND TESTING PLAN OF THE PROPOSED MODEL/SYSTEM

Anaconda is an open-source package manager for Python and R. It is the most popular platform among data science professionals for running Python and R implementations. There are over 300 libraries in data science, so having a robust distribution system for them is a must for any professional in this field. Anaconda simplifies package deployment and management. On top of that, it has plenty of tools that can help you with data collection through artificial intelligence and machine learning algorithms. With Anaconda, you can easily set up, manage, and share Conda environments. Moreover, you can deploy any required project with a few clicks when you're using Anaconda. There are many advantages to using Anaconda and the following are the most prominent ones among them: Anaconda is free and open-source. This means you can use it without spending any money. In the data science sector, Anaconda is an industry staple. It is open-source too, which has made it widely popular. If you want to become a data science professional, you must know how to use Anaconda for Python because every recruiter expects you to have this skill. It is a must-have for data science.

It has more than 1500 Python and R data science packages, so you don't face any compatibility issues while collaborating with others. For example, suppose your colleague sends you a project which requires packages called A and B but you only have package A. Without having package B, you wouldn't be able to run the project. Anaconda mitigates the chances of such errors. You can easily collaborate on projects without worrying about any compatibility issues. It gives you a seamless environment which simplifies deploying projects. You can deploy any project with just a few clicks and commands while managing the rest. Anaconda has a thriving community of data scientists and machine learning professionals who use it regularly. If you encounter an issue, chances are, the community has already answered the same. On the other hand, you can also ask people in the community about the issues you face there, it's a very helpful community ready to help new learners. With Anaconda, you can easily create and train machine learning and deep learning models as it works well with popular tools

including TensorFlow, Scikit-Learn, and Theano. You can create visualizations by using Bokeh, Holo views, Matplotlib, and Data shader while using Anaconda.

How to Use Anaconda for Python

Now that we have discussed all the basics in our Python Anaconda tutorial, let's discuss some fundamental commands you can use to start using this package manager.

Listing All Environments

To begin using Anaconda, you'd need to see how many Conda environments are present in your machine.

conda env list

It will list all the available Conda environments in your machine.

Creating a New Environment

You can create a new Conda environment by going to the required directory and use this command:

```
conda create -n <your_environment_name>
```

You can replace <your_environment_name> with the name of your environment. After entering this command, conda will ask you if you want to proceed to which you should reply with y:

```
proceed ([y])/n)?
```

On the other hand, if you want to create an environment with a particular version of Python, you should use the following command:

```
conda create -n <your_environment_name> python=3.6
```

Similarly, if you want to create an environment with a particular package, you can use the following command:

```
conda create -n <your_environment_name> pack_name
```

Here, you can replace pack_name with the name of the package you want to use.

If you have a .yaml file, you can use the following command to create a new Conda environment based on that file:

```
conda env create -n <your_environment_name> -f <file_name>.yaml
```

We have also discussed how you can export an existing Conda environment to a .yaml file later in this article.

You can activate a Conda environment by using the following command:

```
conda activate <environment_name>
```

You should activate the environment before you start working on the same. Also, replace the term <environment_name> with the environment name you want to activate. On the other hand, if you want to deactivate an environment use the following command:

```
conda deactivate
```

Installing Packages in an Environment

Now that you have an activated environment, you can install packages into it by using the following command:

```
conda install <pack_name>
```

Replace the term <pack_name> with the name of the package you want to install in your Conda environment while using this command.

Updating Packages in an Environment

If you want to update the packages present in a particular Conda environment, you should use the following command:

```
conda update
```

The above command will update all the packages present in the environment. However, if you want to update a package to a certain version, you will need to use the following command:

```
conda install <package_name>=<version>
```

Exporting an Environment Configuration

Suppose you want to share your project with someone else (colleague, friend, etc.). While you can share the directory on GitHub, it would have many Python packages, making the transfer process very challenging. Instead of that, you can create an environment configuration .yaml file and share it with that person. Now, they can create an environment like your one by using the .yaml file.

For exporting the environment to the .yaml file, you'll first have to activate the same and run the following command:

```
conda env export ><file_name>.yaml
```

The person you want to share the environment with only has to use the exported file

by using the 'Creating a New Environment' command we shared before.

Removing a Package from an Environment

If you want to uninstall a package from a specific Conda environment, use the following command:

```
conda remove -n <env_name><package_name>
```

On the other hand, if you want to uninstall a package from an activated environment, you'd have to use the following command:

```
conda remove <package_name>
```

Deleting an Environment

Sometimes, you don't need to add a new environment but remove one. In such cases, you must know how to delete a Conda environment, which you can do so by using the following command:

```
conda env remove --name <env_name>
```

The above command would delete the Conda environment right away.

4.4 PROJECT MANAGEMENT PLAN

Introduction:	October 7-21
Literature Survey:	October 21-23
System Design:	November 1-15
System Implementation:	November 16-29
UML Diagrams:	December 1-25
Conclusions:	January 1-15
References And Future Work:	February 1-20
Testing:	March 1-25

CHAPTER 5

IMPLEMENTATION DETAILS

5.1 Algorithms

5.1.1 Scale -Invariant Feature Transform(SIFT)

The SIFT (Scale-Invariant Feature Transform) algorithm is a computer vision algorithm used for detecting and describing local features in images. It was introduced by David Lowe in 1999 and has since become a popular algorithm in computer vision and image processing.

The SIFT algorithm consists of the following steps:

Scale-space extrema detection: The first step involves identifying potential interest points in the image at different scales using the Difference of Gaussian (DoG) function. The DoG function is obtained by taking the difference of two Gaussian-blurred images at different scales. The local minima and maxima of the DoG function are then identified as potential interest points.

Key point localization: Once the potential interest points are identified, the algorithm refines their location and scale by fitting a 3D quadratic function to the DoG function at each point.

Orientation assignment: The algorithm assigns an orientation to each key point based on the local image gradient directions. This helps in making the descriptor rotationally invariant.

Key point descriptor generation: A descriptor is generated for each key point by considering the local image gradient directions and magnitudes in a surrounding region. This descriptor is a vector of length 128, which is invariant to scale, rotation, and illumination changes.

Key point matching: Finally, the descriptors of two images are compared using a distance metric (e.g., Euclidean distance) to find matching key points between the two images.

The SIFT algorithm has been widely used in various applications, such as image stitching, object recognition, and 3D reconstruction. However, due to its computational complexity, it may not be suitable for real-time applications.

5.1.2 Histogram of Oriented Gradients(HOG)

The "hog algorithm" typically refers to the Histogram of Oriented Gradients (HOG) feature extraction method, which is used in computer vision and image processing for object detection and recognition tasks.

The HOG algorithm works by computing the gradients of an image, which represent the intensity and direction of local edges. These gradients are then quantized into orientation bins, which are used to construct a histogram of gradient orientations within a given image region. The resulting histogram can be used as a feature descriptor to represent the local texture and shape information of an object.

HOG is commonly used in combination with a support vector machine (SVM) classifier to perform object detection and recognition. The SVM classifier is trained on a set of positive and negative examples of the object of interest, using the HOG features as input. During testing, the HOG features of an image are extracted and fed into the SVM, which predicts whether the object is present or not.

HOG has been successfully applied to a wide range of object recognition tasks, including pedestrian detection, face detection, and vehicle detection. However, it is computationally expensive and may not perform well in situations with significant variations in scale and orientation.

5.1.3 Support Vector Machine

Support Vector Machines (SVM) is a supervised machine learning algorithm used for

classification and regression analysis. SVM aims to find the optimal boundary (hyperplane) that separates the data into different classes while maximizing the margin between the classes.

In the case of binary classification, the SVM algorithm works by finding the hyperplane that best separates the data into two classes by maximizing the margin. The margin is defined as the distance between the hyperplane and the nearest data points of each class.

SVM uses a kernel function to map the input data into a high-dimensional feature space where it is easier to find a hyperplane that separates the data. Common kernel functions include linear, polynomial, radial basis function (RBF), and sigmoid.

SVMs are known for their ability to handle high-dimensional data and for their robustness against overfitting. However, SVMs can be sensitive to the choice of kernel function and the regularization parameter.

Overall, SVM is a powerful algorithm that has been widely used in various applications such as image classification, text classification, and bioinformatics.

5.2 Testing

- UNIT TESTING
- INTEGRATION TESTING
- FUNCTIONAL TESTING
- WHITE BOX TESTING
- BLACK BOX TESTING

5.2.1 Unit Testing

- Unit testing is used to ensure that each modular component of the project is working.
- The smallest unit of the software design is the subject of unit testing

- The mentioned project underwent a progressive examination of unit testing.
- The unit testing findings were good and encouraging

5.2.2 Integration Testing

- Integration testing is a methodical methodology for building the software architecture while also running tests to detect faults related to the interface.
- Integration testing, in other words, is the comprehensive testing of the product's collection of modules
- I For the described project, a sequential analysis of Integration Testing was undertaken
- The findings of the integration testing were positive and encouraging

5.2.3 Functional Testing

- Functional test cases included running the code with nominal input values for which the anticipated outputs were known as well as boundary values and unusual values such as logically linked inputs files with identical components, and empty files.
- For the project under consideration, a Functional Testing Sequential Analysis was carried out.
- The outcomes of functional testing were positive and encouraging

5.2.4 White Box Testing

- This kind of testing is also known as glass box testing
- By understanding the precise tasks that a product has been designed to accomplish, testing can be performed to verify that each function is completely operational while also looking for flaws in each function.

- It is a test case design approach that derives test cases from the procedural design's control structure
- White box testing is used for basis path testing
- For the project under consideration ,a sequential assessment of White Box Testing was carried out White Box testing yielded positive and encouraging findings

5.2.5 Black Box Testing

- By understanding a product's internal operation, testing may be carried out to guarantee that "all gears mesh," that is, the internal operation operates according to 25 specifications and all internal components have been appropriately exercised
- For the described project, a sequential analysis of Black Box Testing was undertaken
- The outcomes of functional testing were positive and encouraging.

CHAPTER 6

RESULTS

The online hand gesture recognition and classification system will be able to recognize a range of hand gestures in real-time. The system will be able to translate the recognized gestures into text or speech, depending on the user's preferences.

The system will be tested using a range of hand gestures, and the accuracy of the system will be evaluated. The accuracy of the system will be measured in terms of the percentage of correctly recognized gestures.

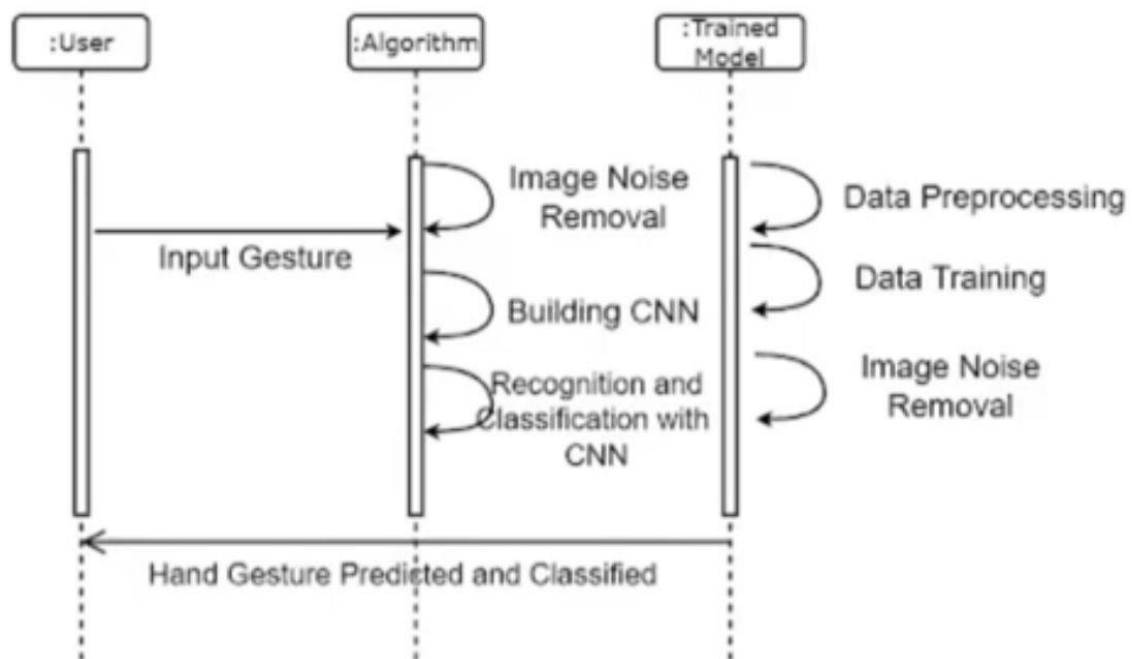


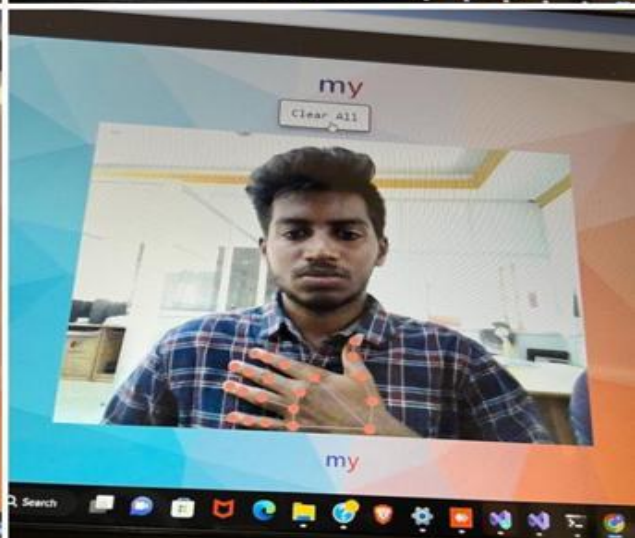
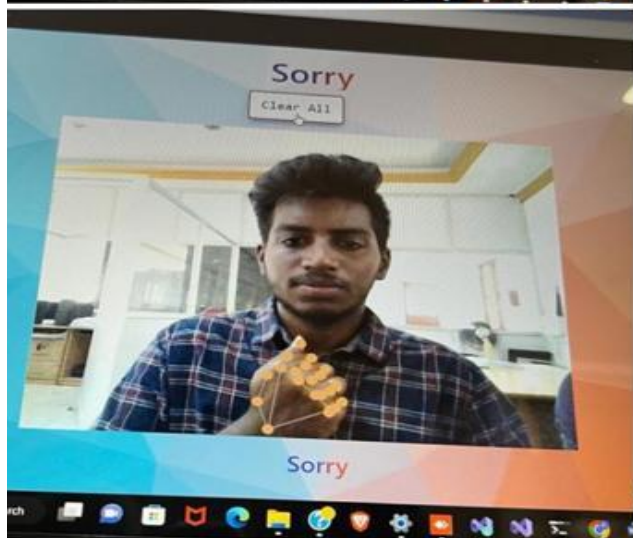
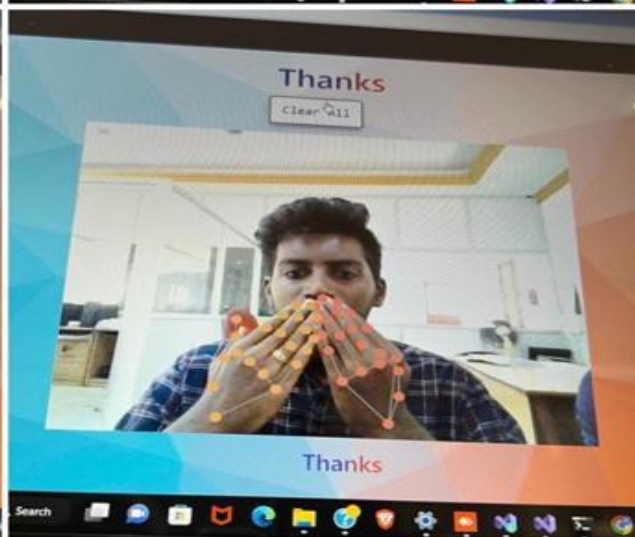
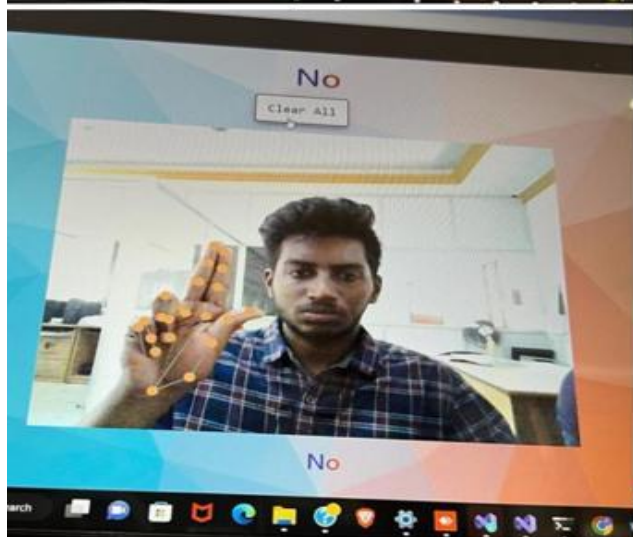
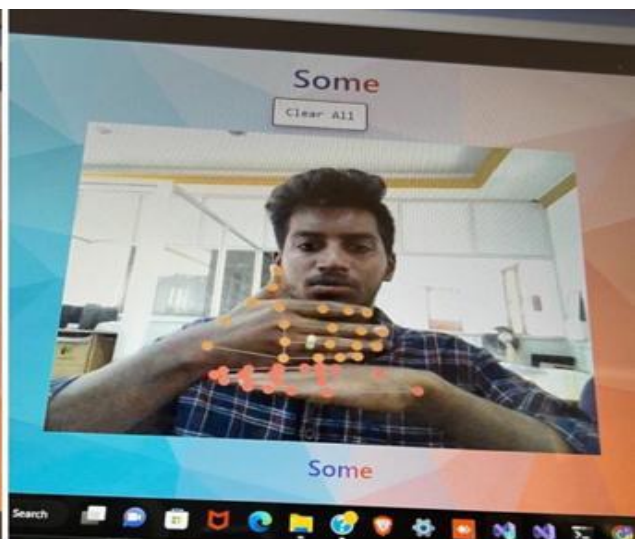
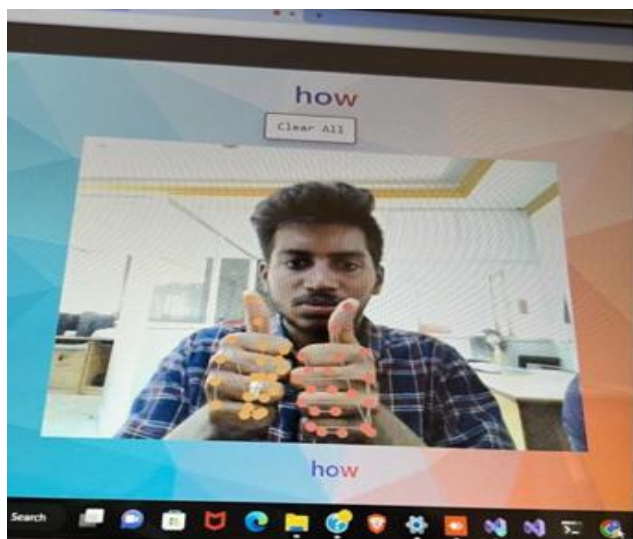
Figure 6.1 Sequence Diagram



Figure 6.2 Data Flow Diagram

In this work, a deep learning based CNN model has been proposed to identify static hand gestures. Traditional models have been found out to be restrictive in the sense that they need preprocessing of the image to achieve good accuracy. However, deep learning models have advantage over traditional methods in that they don't need prior feature extraction. Since they do not involve any feature extraction they are very robust. Since we employed DCNN model it is very attractive for gesture recognition applications. The model thus trained has shown good results with cluttered backgrounds, poor lighting conditions, and different hand morphologies. The robustness of the model has been attributed to the capacity of DCNNs to learn complex features in images.

Another important feature about the present model is that it is very accurate. The model has achieved 99.6% test accuracy which is one of the highest ever reported test accuracy. High accuracy can be attributed to careful data collection, data augmentation and network architecture. While collecting data, care has been taken to include various scenarios into account. Good data collection coupled with data augmentation led to generalization of the network.



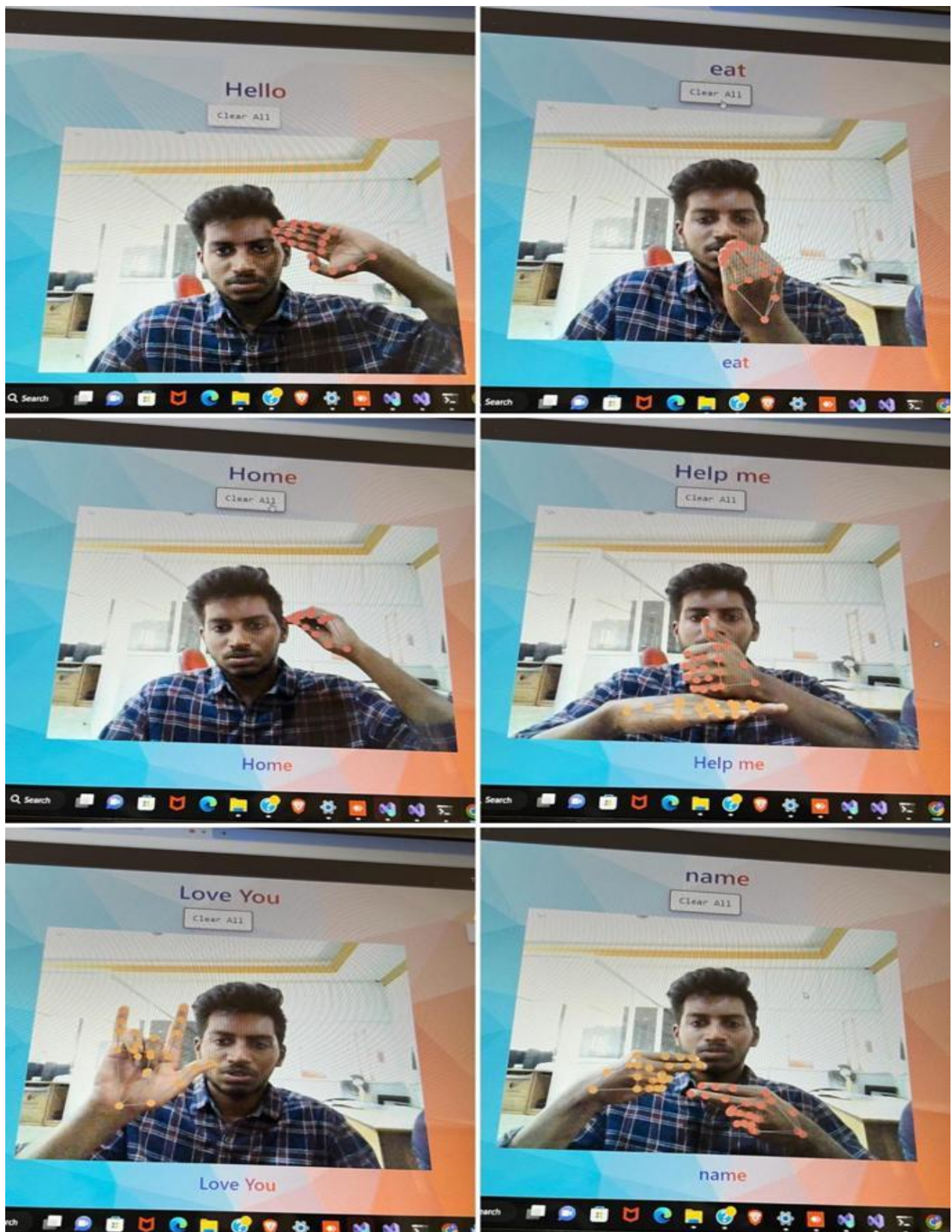


Figure 6.3 Output Screenshot

CHAPTER 7

7.1 Conclusion:

This project introduces a CNN-based approach for the recognition and classification of sign language using deep learning .Our proposed system was able to achieve 99% training accuracy with testing accuracy of 90.04% in letter recognition, 93.44% in number recognition, and 97.52% in static word recognition, obtaining an average of 93.667% based on the gesture recognition with limited time. Ag for future work, the evolution of the proposed system can be generalized for a wider class of hand gestures. Furthermore, the impact of different parameters on the performance of the system and the accuracy of results can be further investigated.

Time frame selection techniques can be bettered, more optimization can be added, and the loss function can be studied more deeply Using open CV, we offer a technique for identifying and recognizing signals in this paper. Throughout the training of each system, each letter, number and word gesture were utilized 50 times. This method has substantially better accuracy and significantly fewer false positives than the others. The identification of hand gestures has seen significant advancement in the last several decades thanks to the efforts of numerous experts. Sign language interpretation devices, augmented or virtual reality, robot control, and the ability to recognize hand gestures are just a few of the numerous areas where this technology has proven useful. Since the introduction of the next generation of gesture interface technology, the value of gesture recognition has skyrocketed. Most people who use sign language are deaf or mute and hence unable to communicate verbally. It is difficult to express emotions clearly when communicating with someone who can see but do not share a common hand sign language.

OpenCV provides a fast and convenient way of recognizing and classifying hand gestures. This is especially suitable for the Deaf and Dumb community, who have limited access to other forms of communication. The cost of implementation is also quite low, making it an ideal solution. In conclusion, hand gesture recognition and classification for deaf and dumb can bring about transformative and life-changing changes in the lives of people with hearing impairments. With the use of machine learning and open source

libraries, it is becoming increasingly easier to develop projects centered around hand gesture recognition and classification.

7.2 Future Enhancements

As for future work, the evolution of the proposed system can be generalized for a wide class of hand gestures. Furthermore, the impact of different parameters on the performance of the system and the accuracy of results can be further investigated. Time frame selection techniques can be bettered, more optimization can be added, and the loss function can be studied more deeply.

7.3 Implementation

- ***Image Pre- Processing***

The captured objects are being pre-processed. Pre-processing is done to remove the extra objects.

- ***Feature Extraction and Recognition***

An algorithm is used to determine an image's characteristics. To retrieve the best featured image from the database, the algorithm applied to the captured image.

REFERENCES:-

- [1] Soma Shrenika, Myneni Madhu Bala, "Sign Language Recognition Using Template Matching Technique", International Conference on Computer Science Engineering and Applications (ICCSEA), 2020
- [2] H Muthu Mariappan, V Gomathi, "Real-Time Recognition of Indian Sign Language", International Conference on Computational Intelligence in Data Science (ICCIDS), 2019
- [3] Wanbo Li, Hang Pu, Ruijuan Wang, "Sign Language Recognition Based on Computer Vision", IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA), 2021
- [4] Pushkar Kurhekar, Janvi Phadtare, Sourav Sinha, Kavita P. Shirsat, "Real Time Sign Language Estimation System", 3rd International Conference on Trends in Electronics and Informatics (ICOEI), 2019
- [5] Aniket Kumar, Mehul Madaan, hubham Kumar, Aniket Saha, Suman Yadav, "Indian Sign Language Gesture Recognition in Real-Time using Convolutional Neural Networks", 8th International Conference on Signal Processing and Integrated Networks (SPIN), 2021
- [6] Sai Nikhilesh Reddy Karna, Jai Surya Kode, Suneel Nadipalli, Sudha Yadav, "American Sign Language Static Gesture Recognition using Deep Learning and Computer Vision", 2nd International Conference on Smart Electronics and Communication (ICOSEC), 2021
- [7] Shirin Sultana Shanta, Saif Taifur Anwar, Md. Rayhanul Kabir, "Bangla Sign Language Detection Using SIFT and CNN", 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT), 2018
- [8] Rajiv Ranjan, B Shivalal Patro, Mohammad Daim Khan, Manas Chandan Behera, Raushan Kumar, Utsav Raj, "A Review on Sign Language Recognition Systems", IEEE 2nd International Conference on Applied Electromagnetics, Signal Processing, & Communication (AESPC), 2022

- [9] Kohsheen Tiku, Jayshree Maloo, Aishwarya Ramesh, Indra R, "Real-time Conversion of Sign Language to Text and Speech", Second International Conference on Inventive Research in Computing Applications (ICIRCA), 2020
- [10] Vishwa Hariharan Iyer, U.M Prakash, Aashrut Vijay, P. Sathishkumar, "Sign Language Detection using Action Recognition", 2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACIT)

