TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

SAGARMATHA ENGINEERING COLLEGE

A PROJECT REPORT

ON

"NEPALI SIGN LANGUAGE RECOGNITION USING CNN"

SUBMITTED BY:

ANISH KUMAR SHAH	(39852)
BISHAL KARKI	(39854)
ROSITA TAMANG	(39861)
YUKI THAPA	(39869)

A PROJECT REPORT SUBMITTED TO THE DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF BACHELOR IN COMPUTER ENGINEERING

DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING

SANEPA, LALITPUR, NEPAL

MAY 3, 2023

NEPALI SIGN LANGUAGE RECOGNITION USING CNN

SUBMITTED BY

Anish Kumar Shah	(39852)
Bishal Karki	(39854)
Roshita Tamang	(39861)
Yuki Thapa	(39869)

Project Supervisor

Er. Sujan Pokhrel

A Project Submitted to the Department of Electronics and Computer Engineering in partial fulfilment of the requirements for the degree of Bachelor in Computer Engineering

Department of Electronics and Computer Engineering

Institute of Engineering, Sagarmatha Engineering College

Tribhuvan University Affiliate

Sanepa, Lalitpur, Nepal

MAY 3, 2023

COPYRIGHT ©

DECLARATION

We hereby declare that the report of the project work entitled **"Nepali Sign Language Recognition Using CNN"**which is being submitted to the Sagarmatha Engineering College, IOE, Tribhuvan University, in the partial fulfillment of the requirements for the award of the Degree of Bachelor of Engineering in Computer Engineering, is a bonafide report of the work carried out by us.The material contained in this report has not been submitted to any University or Institution for the award of any degree.

Group Members:

Anish Kumar Shah (39852)

Bishal Karki (39854)

Rosita Tamang (39861)

Yuki Thapa (39869)

CERTIFICATE OF APPROVALS

The undersigned certify that they have read and recommended to the Department of Electronics and Computer Engineering for acceptance, a project work entitled **"NEPALI SIGN LANGUAGE RECOGNITION USING CNN"**, submitted by **Anish Kumar Shah, Bishal Karki, Rosita Tamang and Yuki Thapa** in partial fulfillment of the requirement for the degree of "Bachelor in Computer Engineering".

.....

External Examiner: Ananda Raj Khanal
Former Senior Director
Nepal Telecommunication Authority

.....

Supervisor: Er. Sujan Pokhrel
Lecturer, Department of Electronics and Computer Engineering,
Sagarmatha Engineering College
Tribhuvan University Affilate
Senepa,Lalitpur

.....

Project Coordinator: Er. Bipin Thapa Magar
Lecturer, Department of Electronics and Computer Engineering,
Sagarmatha Engineering College
Tribhuvan University Affilate
Senepa,Lalitpur

DEPARTMENTAL ACCEPTANCE

The project work entitled " **NEPALI SIGN LANGUAGE RECOGNITION USING CNN**", submitted by **Anish Kumar Shah, Bishal Karki, Rosita Tamang,Yuki Thapa** in partial fulfillment of the requirement for the award of the degree of **"Bachelor of Engineering in Computer Engineering"** has been accepted as a bonafide record of work independently carried out by team in the department.

.....

Senior Lecturer: Er. Bharat Bhatta

Head of Department

Department of Electronics and Computer Engineering,

Sagarmatha Engineering College,

Tribhuvan University Affiliate,

Sanepa, Lalitpur

Nepal.

ACKNOWLEDGEMENT

Foremost, we would like to express our sincere gratitude to **Er. Bharat Bhatta, Head Of Department(HOD)** and **Er. Baikuntha Kumar Acharya, Deputy Head Of Department(DHOD)** of Department of Computer and Electronics Engineering, Sagarmatha Engineering College for providing us an opportunity to work on our major project and providing proper suggestions and allowing us to use facilities available.

In addition to our mentor, we would like to thank **Er. Sujan Pokhrel** and project coordinator **Er. Bipin Thapa Magar** for their guidance,suggestions and for directly supporting us in tackling various difficulties.

We are also indebted to all the faculty members of the Department of Electronics and Computer engineering for support,advice, help, and suggestion during this project. Our thanks and appreciation also go to all our teachers, class friends as well as all our well-wishers who have willingly helped us out with their knowledge and support.

ABSTRACT

People with speech or hearing impairments, including mute or deaf individuals, heavily rely on visual communication.To bridge this gap and enable two-way communication, it is necessary to build a system that can translate sign language into text.With this in mind, we develop a Nepali Sign Language Recognition system using Convolutional Neural Networks (CNN) to improve accessibility and communication for the deaf and hard of hearing community in Nepal.However, due to the unavailability of a suitable datasets, 36 different sign language of Nepali consonants datasets were custom made, covering a wide range of hand gestures and movements commonly used in Nepali Sign Language.The CNN model was trained and optimized using these datasets,achieving high accuracy of 98.64% in recognizing different signs.The system was also evaluated on a test dataset, demonstrating promising results.The system aims to break the communication barrier between able and disabled individuals while contributing to the Nepali datasets.

Keywords: Convolution neural network, ReLU, datasets, sign-language, NSL, TensorFlow Deep Learning.

TABLE OF CONTENTS

COPYRIGHT ii

DECLARATION iii

RECOMMENDATION iv

DEPARTMENTAL ACCEPTANCE v

ACKNOWLEDGEMENT vi

ABSTRACT vii

TABLE OF CONTENTS viii

LIST OF FIGURES x

LIST OF TABLES xi

LIST OF ABBREVIATIONS xii

1 INTRODUCTION 1

1.1 Background 2

1.2 Problem definition 3

1.3 Objectives 3

1.4 Significance 3

1.5 Limitations of the Project Work 3

2 LITERATURE REVIEW 5

3 RELATED THEORY 8

3.1 Deep Learning 8

3.1.1 Convolution Neural Network (CNN) 9

3.1.2 Layers of CNN 9

3.1.3 Technologies 11

4 METHODOLOGY **13**

4.1 System Block Diagram 13

4.2 System Architecture 14

4.3 Flowchart 15

4.4 Use Case Diagram 16

4.5 Data Flow Diagram 17

4.6 Sequence Diagram 19

4.7 State Chart Diagram 20

4.8 Class Diagram 21

4.9 Software Development Model 22

4.10 Techniques 23

4.10.1 Dataset Preparation 23

4.10.2 Image Preprocessing 23

4.10.3 Splitting of Dataset 25

4.10.4 Training of Convolutional Neural Network model 25

5 RESULT AND ANALYSIS **28**

5.1 Result 28

5.1.1 Output and Analysis 28

6 EPILOGUE **33**

6.1 Conclusion 33

6.2 Future Enhancement 33

REFERENCES **35**

APPENDIX **36**

LIST OF FIGURES

Figure 1.1 Nepali Sign Language 2

Figure 3.1 Basic Outline of a Convolution Neural Network 9

Figure 3.2 Min and Average Polling 10

Figure 3.3 Fully Connected layer 11

Figure 4.1 System Block Diagram of Nepali Sign Language Recognition 13

Figure 4.2 System Architecture of Nepali Sign Language Recognition . 14

Figure 4.3 System Flowchart of Nepali Sign Language Recognition . . . 15

Figure 4.4 Use Case Diagram of Nepali Sign Language Recognition . . 16

Figure 4.5 DFD Level 0 Diagram of Nepali Sign Language Recognition 17

Figure 4.6 DFD Level 1 Diagram of Nepali Sign Language Recognition 18

Figure 4.7 Sequence Diagram of Nepali Sign Language Recognition . . 19

Figure 4.8 State Chart Diagram of Nepali Sign Language Recognition . 20

Figure 4.9 Class Diagram of Nepali Sign Language Recognition 21

Figure 4.10 Iterative Model 22

Figure 4.11 Snapshot of RGB to Gray-Scale Image 23

Figure 4.12 Snapshot of Blurring of Gray-Scale Image 24

Figure 4.13 Snapshot of Threshold Image 25

Figure 4.14 Layers of CNN 27

Figure 5.1 Accuracy Graph 29

Figure 5.2 Loss Graph 30

Figure 5.3 Loss VS Accuracy Graph 30

Figure 5.4 Confusion Matrix of Sign Language Detection 31

LIST OF TABLES

Table 5.1 Result Table 28
Table 5.2 Training History 29

LIST OF ABBREVIATIONS

AI	Artificial Intelligence
ASL	American Sign Language
CNN	Convolution Neural Network
DFD	Data Flow Diagram
NSL	Nepali Sign Language
RELU	Rectified Linear Unit
RGB	Red Green Blue
SGD	Stochastic Gradient Descent
HCI	Human Computer Interaction

CHAPTER 1

INTRODUCTION

Gesture recognition has emerged as a promising field in Human-Computer Interaction (HCI), allowing for intuitive and interactive communication with computers. Nepali Sign Language (NSL) is a standardized language used by the hearing-impaired community in Nepal. NSL uses dynamic and static finger-spelling and a single hand to represent 49 alphabet sets, including 36 consonants and 13 vowels. In this project, we developed a system to recognize Nepali sign language gestures and translate them into text, thus enabling two-way communication between individuals with speech and hearing impairments and those without. The system uses vision-based technology and creates datasets by extracting images from videos. Our aim is to contribute to the Nepali datasets and bridge the communication gap between the able and disabled communities. .[1]

Figure 1.1: Nepali Sign Language

1.1 Background

The prevalence of deafness in Nepal is approximately 2.8 percent, accounting for approximately 7 lakh people. Approximately 10 percentage of them are oblivious to their problem due to mild impairment and 7 percent of them have a disabling impairment. There are approximately 200 different sign languages. The vast majority of deaf persons in Nepal do not speak Nepalese Sign Language as a first language. Nepalese Sign Language is largely taught in schools for the deaf because the vast majority of deaf children there, like in all other countries, are born into hearing families with no one who signs. The majority of deaf Nepalese, however, do not get the opportunity to learn Nepalese Sign Language normally because these schools are small in number and not easily accessible to the majority.

1.2 Problem definition

Because only a small percentage of gestures are recognized by most people, people with hearing loss are typically deprived of the ability to connect with others in a general way. Additionally, most people with disabilities utilize sign language, which is difficult for the general public to understand. It is necessary to create a system where both parties can comprehend one another in order for communication between people with disabilities and the broader population to be productive.

1.3 Objectives

The Objectives of our project is as follows:

- To convert sign languages used for Nepali Alphabets (Consonants) into corresponding text alphabets.

1.4 Significance

The Significance of our project is as follows:

- The project can increase accessibility to various services and opportunities for deaf or hard-of-hearing individuals who use Nepali sign language.
- The project can help bridge the communication gap between deaf or hard-of-hearing individuals who use Nepali sign language and those who do not.
- The project can support education and learning for Nepali sign language users, as well as promote the preservation and development of the language.

1.5 Limitations of the Project Work

The limitations of our project is as follows:

- It only recognizes 36 different Nepali Alphabets (consonant) sign language and gives it corresponding text alphabets as output.

- The environment should be bright and lit while giving the input.
- It only takes single sign as an input and gives its corresponding single text alphabet as an output.

CHAPTER 2

LITERATURE REVIEW

They propose a fast-multi-scale feature detection and description method for hand gesture recognition.Where they firstly approximate complex Gaussian derivatives with simple integral images in feature detection. Then multiscale geometric descriptors at feature points are obtained to represent hand gestures. Finally, the gesture is recognized with its geometric configuration. Specific gesture is required to trigger the hand detection followed by tracking; the hand is further segmented using motion and color cues [2].

The authors(R.cutlur and M.TURK) have developed a real-time, view-based gesture recognition system. Optical flow is estimated and segmented into motion blobs. Gestures are recognized using a rule-based technique based on characteristics of the motion blobs such as relative motion and size. Parameters of the gesture (e.g., frequency) are then estimated using context specific techniques. The system has been applied to create an interactive environment for children [3].

This paper have described a system for automatic real-time control of a windowed operating system entirely based on one-handed gesture recognition. Using a computer-mounted video camera as the sensor, the system can reliably interpret a 46-element gesture set at 15 frames per second on a 600MHz notebook PC. The gesture set, comprises the 36 letters and digits of the American Sign Language finger spelling alphabet three 'mouse buttons', and some ancillary gestures. The accompanying video show a transcript of about five minutes of system operation in which files are created, renamed, moved,and edited—entirely under gesture control. This corresponds to a per-image recognition rate of over 99 percent [4].

A new hand gesture recognition method based on Input– Output Hidden Markov Models is presented. This method deals with the dynamic aspects of gestures. Gestures were 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 main goal of this system was to recognize two classes of gestures: deictic and symbolic [5].

In this paper, Augmented desk interfaces and other virtual reality systems depend on accurate, real-time hand and fingertip tracking for seamless integration between real objects and associated digital information. We introduce a method for discerning fingertip locations in image frames and measuring fingertip trajectories across image frames. We also propose a mechanism for combining direct manipulation and symbolic gestures based on multiple fingertip motions.Our method uses a filtering technique, in addition to detecting fingertips in each image [6].

Using orientation histograms a simple and fast algorithm would have developed to work on a workstation. It will recognize static hand gestures, namely, a subset of American Sign Language (ASL).Previous systems have used data gloves or markers for input in the system.A pattern recognition system have used a transform to convert an image into a feature vector, which would compared with the feature vectors of a training set of gestures. The final system was implemented with a Perceptron network [7].

Image classification, many scholars have proposed their methods. Pang et al. proposed a novel fused CNN that fused the features extracted in the shallow and deep layers for the biomedical image classification Ivan et al. studied an image classification method on a tight representation budget, it focused on very short image descriptor which might be lost during the training process Zhou et al. proposed a novel data augmentation strategy for the Siamese model and introduced a joint decision mechanism into the model which can better improve the classification performance . All the above methods are well-behaved in the image classification filed and widely applied in many different fields. However, the structures of these CNN models are very complex, and the consumption of computing resources and the reduction of efficiency cannot be overlooked. Therefore, it is necessary to propose a more efficient image classification method.[8]

A convolutional neural network is composed of alternatively stacked convolutional layers and spatial pooling layers. The convolutional layer is to extract feature maps by linear convolutional filters followed by nonlinear activation functions (e.g., rectifier, sigmoid, tanh, etc.). Spatial pooling is to group the local features together

from spatially adjacent pixels, which is typically done to improve the robustness to slight deformations of objects. CNNs have been long studied and applied in the field of computer vision. More than a decade ago, LeCun et al. trained multilayer neural networks with the back-propagation algorithm and the gradient learning technique, and then demonstrated its effectiveness on the handwritten digit recognition task. The deep CNNs have exhibited good generalization power in image-related applications. Recently, Krizhevsky et al. achieved a breakthrough, outperforming the existing handcrafted features on ILSVRC 2012 which contains 1000 object classes. Since 2012, CNNs have draw a resurgence of attention in various visual recognition tasks such as image classification semantic segmentation object recognition video analysis etc. Recently the networks are going deeper, such as GoogLeNet which won 2014 ILSVRC classification challenge by employing 22 layers. In the number of layers of the proposed residual nets reaches to 152 and achieves better performance.There are some deep learning related works on HSI classification in the literature. Such as in deep stacked autoencoders are employed to extract features. The autoencoder is a kind of unsupervised method. The proposed method in combines principle component analysis (PCA),autoencoders and logistic regression, and it is not an end-to-end deep method. An end-to-end deep CNN method is proposed in . The method takes the raw data as the input and outputs the predicted class labels. The number of training samples of each class is 200, and it is a relative large number.[9]

CHAPTER 3

RELATED THEORY

3.1 Deep Learning

Deep learning is a machine learning technique that teaches computers to do what comes naturally to humans: learn by example.In deep learning, a computer model learns to perform classification tasks directly from images, text, or sound. Deep learning models can achieve state-of-the-art accuracy, sometimes exceeding human-level performance. Models are trained by using a large set of labeled data and neural network architectures that contain many layers. It is of two types supervised and unsupervised learning.

a. Supervised Learning

   Supervised learning, as the name indicates, has the presence of a supervisor as a teacher. Basically supervised learning is when we teach or train the machine using data that is well labelled. Which means some data is already tagged with the correct answer.

b. Unsupervised Learning

   Unsupervised learning is the training of a machine using information that is neither classified nor labeled and allowing the algorithm to act on that information without guidance. Here the task of the machine is to group unsorted information according to similarities, patterns, and differences without any prior training of data.

c. Reinforcement Learning

   Reinforcement learning (RL) is an area of machine learning concerned with how intelligent agents ought to take actions in an environment in order to maximize the notion of cumulative reward. Reinforcement learning is one of three basic machine learning paradigms, alongside supervised learning and

unsupervised learning.

3.1.1 Convolution Neural Network (CNN)

A convolutional neural network (CNN or ConvNet), is a network architecture for supervised deep learning which learns directly from data, eliminating the need for manual feature extraction.CNNs are particularly useful for finding patterns in images to recognize objects, faces, and scenes. They can also be quite effective for classifying non-image data such as audio, time series, and signal data.Applications that call for object recognition and computer vision — such as self-driving vehicles and face-recognition applications — rely heavily on CNNs.[10]

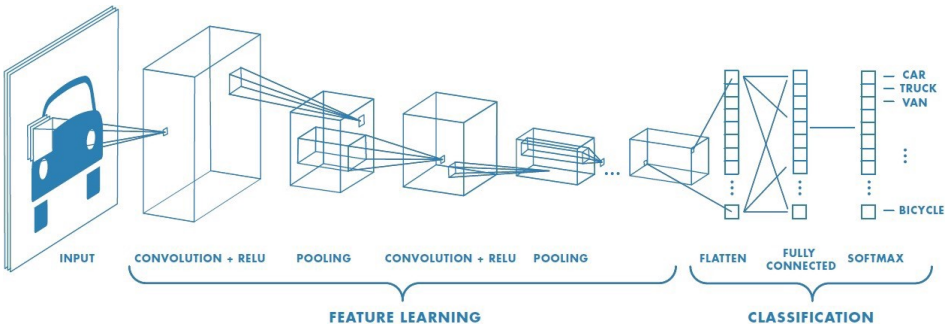


Figure 3.1: Basic Outline of a Convolution Neural Network

3.1.2 Layers of CNN

a. Convolutional Layer

The convolutional layer is the core building block of a CNN, and it is where the majority of computation occurs. It requires a few components, which are input data, a filter, and a feature map. Let's assume that the input will be a color image, which is made up of a matrix of pixels in 3D. This means that the input will have three dimensions—a height, width, and depth—which correspond to RGB in an image. We also have a feature detector, also known as a kernel or a filter, which will move across the receptive fields of the image, checking if the feature is present. This process is known as a convolution.

b. Pooling Layer

Pooling layers, also known as downsampling, conducts dimensionality reduction, reducing the number of parameters in the input. Similar to the convolutional layer, the pooling operation sweeps a filter across the entire input, but the difference is that this filter does not have any weights. Instead, the kernel applies an aggregation function to the values within the receptive field, populating the output array. There are two main types of pooling:

a. Max pooling:

As the filter moves across the input, it selects the pixel with the maximum value to send to the output array. As an aside, this approach tends to be used more often compared to average pooling.

b. Average pooling:

As the filter moves across the input, it calculates the average value within the receptive field to send to the output array.[10]

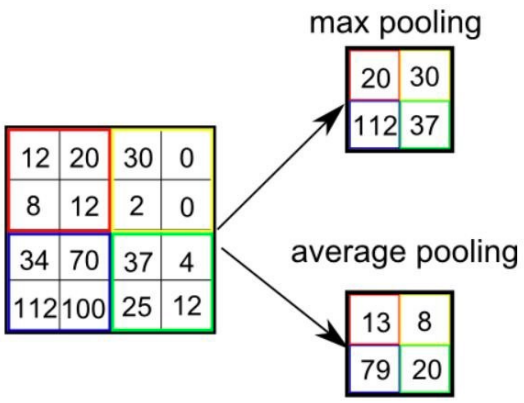


Figure 3.2: Min and Average Polling

c. Fully-Connected Layer

The name of the full-connected layer aptly describes itself. As mentioned earlier, the pixel values of the input image are not directly connected to the output layer in partially connected layers. However, in the fully-connected layer, each node in the output layer connects directly to a node in the previous layer.This layer performs the task of classification based on the features extracted through the previous layers and their different filters.

While convolution and pooling layers tend to use ReLU functions, FC layers usually leverage a softmax activation function to classify inputs appropriately, producing a probability from 0 to 1.[10]

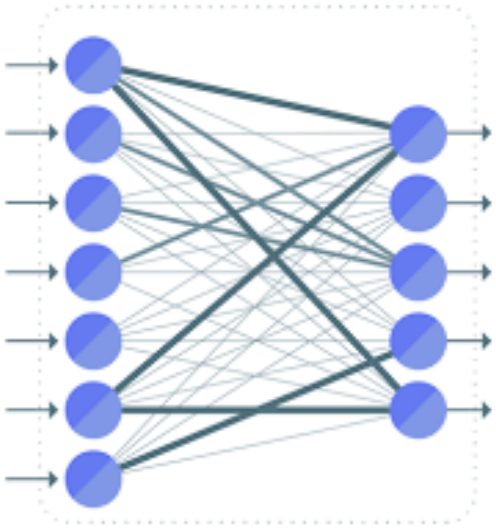


Figure 3.3: Fully Connected layer

3.1.3 Technologies

a. Python

Python is an interpreter, high-level and general purpose programming language. Python's design system indicates readability of code with its notable use of important whitespace. Its language constructs and object-oriented way aims to help programmers write clear, legitimate code for small and large-scale projects.

b. Numpy

NumPy is a Python library used for working with arrays. It also has functions for working in the dominion of linear algebra, Fourier transform, and matrices. NumPy was created in 2005 by Travis Oliphant. NumPy stands for Numerical Python.

c. OpenCV

OpenCV is a vast open-source library for computer vision, machine learning,

and image processing. OpenCV supports a large variety of programming languages like Java, C++, Python, etc. It can process videos and images to identify objects, faces, or even the handwriting of a human. When it is unified with many diverse libraries,such as Numpy which is a highly developed library for numerical operations, then the number of weapons increases in your Arsenal i.e., whatever operations that can be done in Numpy can also be combined with OpenCV. The OpenCV tutorial will help you learn the Image processing from Basics to Advance, like operations on Images, Videos using a huge set of OpenCV programs and projects.

d. Keras

Keras is a high-level neural networks library that is running on the top of TensorFlow, CNTK, and Theano. Using Keras in deep learning allows easy and fast prototyping as well as running seamlessly on CPU and GPU. This framework is written in Python coding language which is easy to debug and allows ease for resilience.

e. TensorFlow

TensorFlow is an end-to-end open-source platform for machine learning. It's a comprehensive and flexible ecosystem of tools, libraries and other resources that provide workflow with high-level APIs. The framework offers many different levels of concepts for you to choose the one you need to build and deploy machine learning models.

CHAPTER 4

METHODOLOGY

4.1 System Block Diagram

The block diagram of our system is as follows:

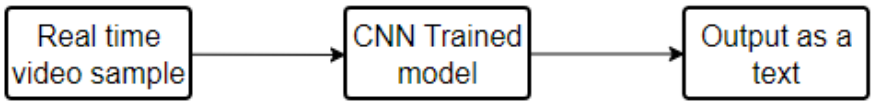


Figure 4.1: System Block Diagram of Nepali Sign Language Recognition

The above diagram shows the block diagram of our system. Here the video from the webcam is taken as input. It is then pre-processed and stored as frame which is then compare with the trained model which we have trained using the sets of datasets we have created and then it gives the output in the form of text.

4.2 System Architecture

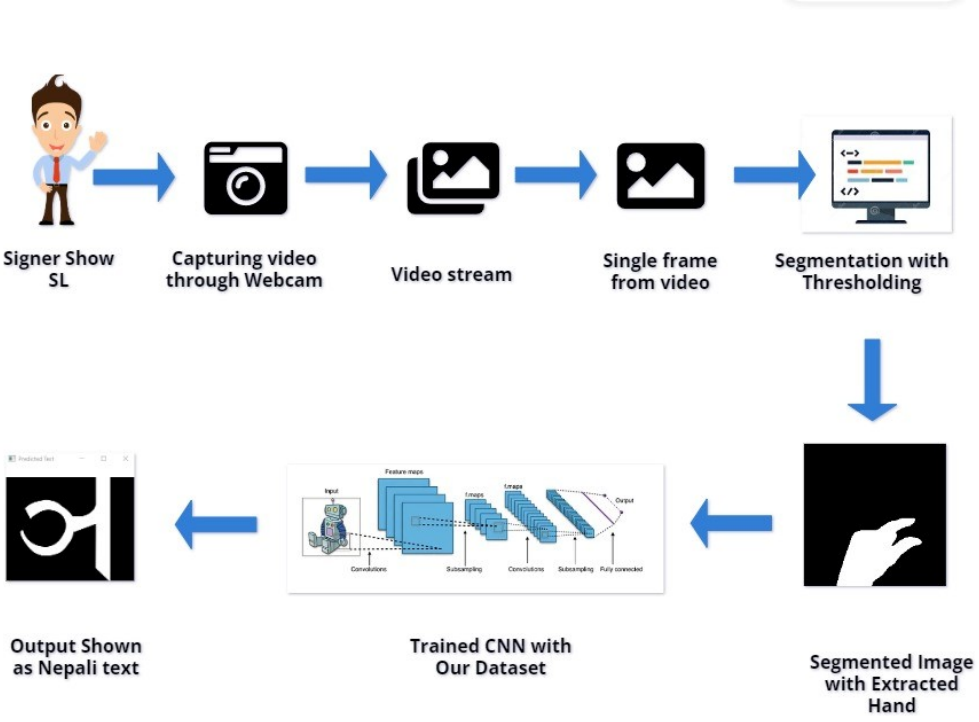


Figure 4.2: System Architecture of Nepali Sign Language Recognition

The above diagram shows the diagram of proposed methodology of Nepali Sign Language Recogination system using CNN. Here the signer performs different nepali sign langauge and the real time video is capture through the camera or an webcam. The video is converted into frames. Segmentation is done along with thresholding which seprates the background of our input. Then the segmented image with extracted hand is passed through the model trained with CNN using our own dataset and then the ouput is shown in nepali text to the user.

4.3 Flowchart

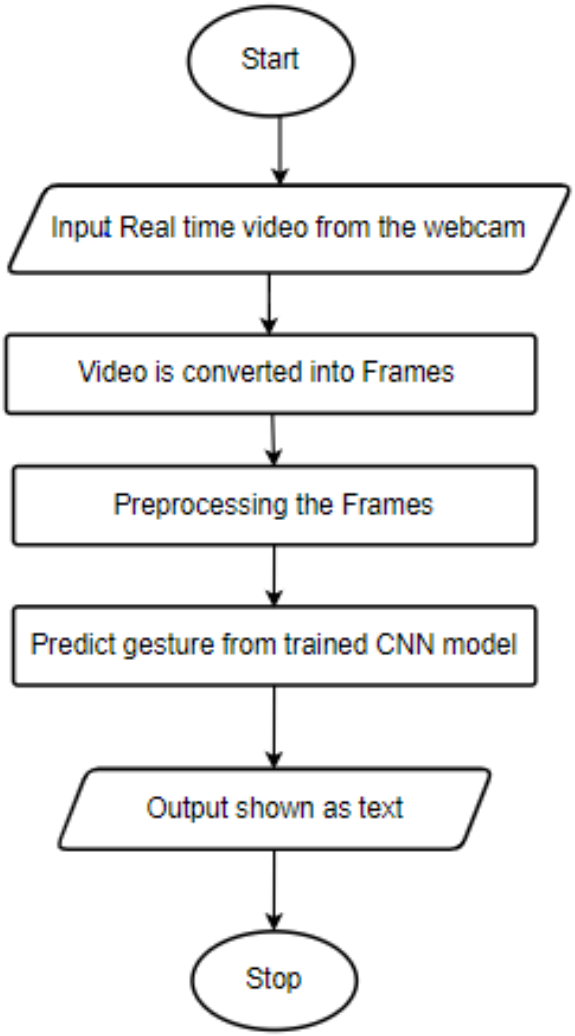


Figure 4.3: System Flowchart of Nepali Sign Language Recognition

The above figure shows the flowchart of our system ,where the process start by taking the real time video from the webcam and the capture video converted into frames. The frames is preprocess using a multiple steps such as bluring ,grayscale conversion,thresholding ,etc .Then finally it is fed as input to our trained CNN model and predict gesture . The final output is shown as in a text format.

4.4 Use Case Diagram

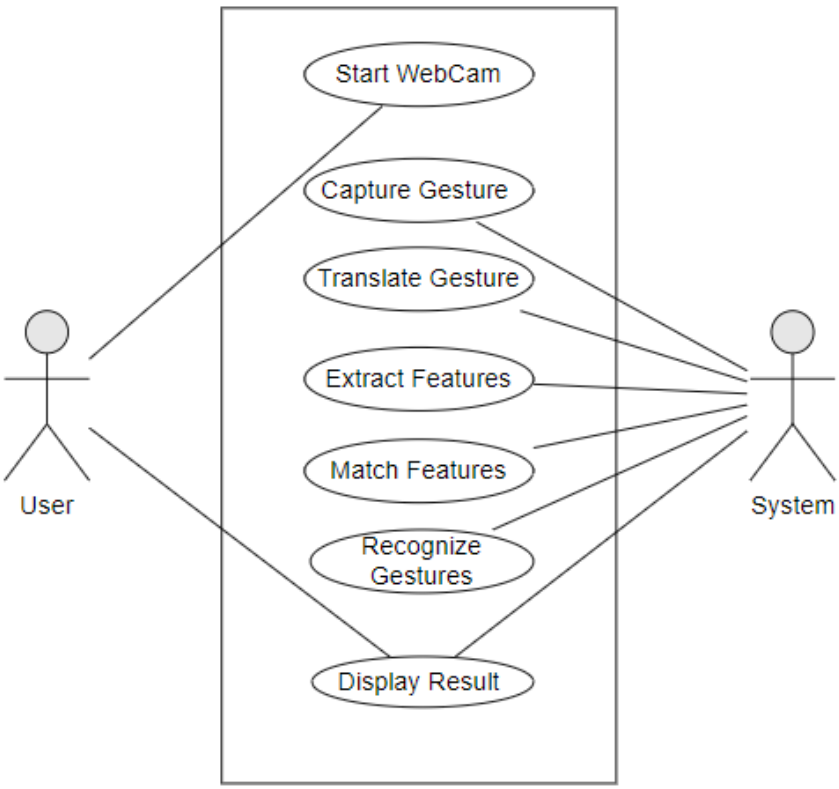


Figure 4.4: Use Case Diagram of Nepali Sign Language Recognition

The above figure shows the Use Case Diagram where the user will input the Nepali sign language after the webcam opens and the system will capture the gesture, translate the gesture and match features to the trained model. Then it recognize the gestures and display the result to the user.

4.5 Data Flow Diagram

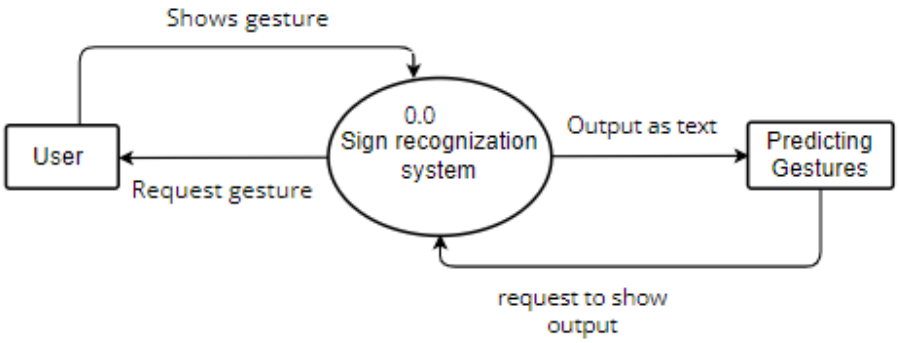


Figure 4.5: DFD Level 0 Diagram of Nepali Sign Language Recognition

The above figure shows the DFD level 0 diagram where there are three entities. The video is feed to the system through webcam. Sign recognition recognize the image and predict the label. And the output is display to the user.

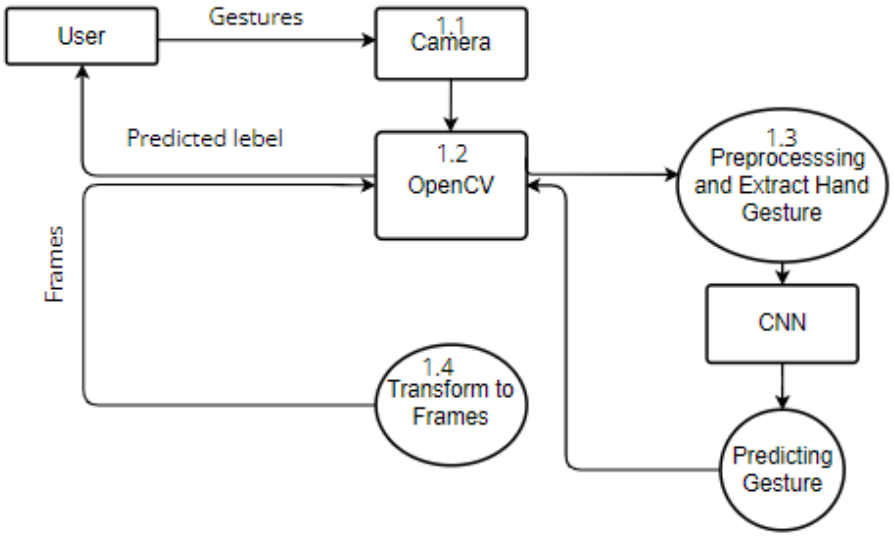


Figure 4.6: DFD Level 1 Diagram of Nepali Sign Language Recognition

The above figure shows the DFD level 1 diagram. User shows the sign on the camera which operate with the help of OpenCV. The video from camera or webcam is transform into frames. The hand is extracted from the frames and pass to the trained mode, trained using Convolutional Neural Network(CNN) which predict the sign(gesture) and through OpenCV the predicted label is display to the user.

4.6 Sequence Diagram

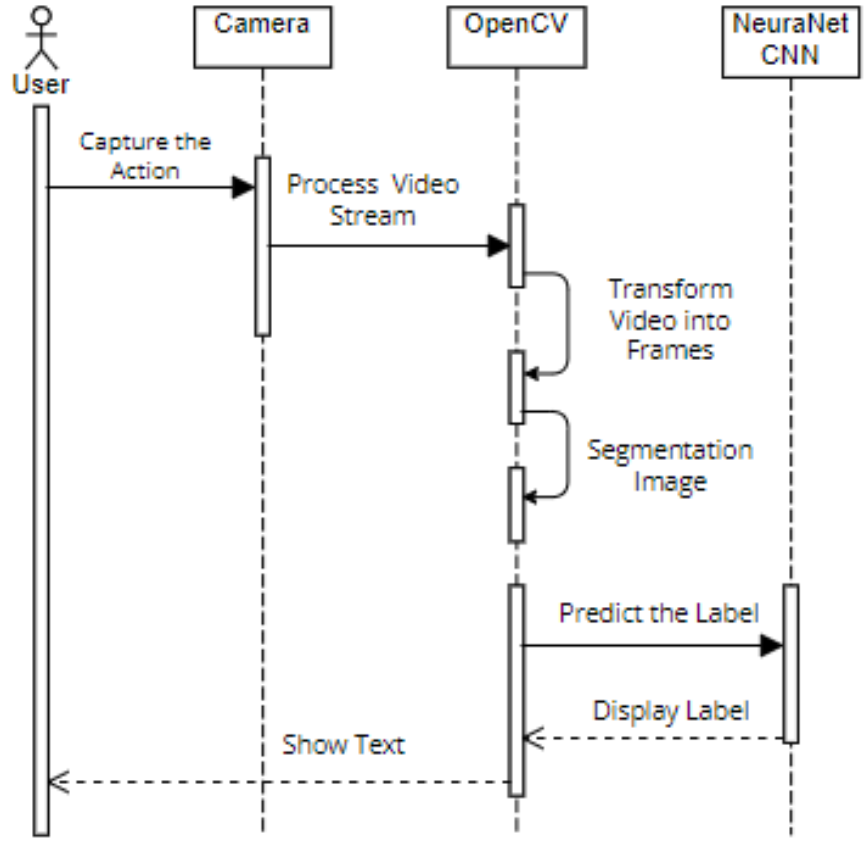


Figure 4.7: Sequence Diagram of Nepali Sign Language Recognition

The above figure shows the sequence diagram where user provide action which is the video of user performing sign is captured using camera. The camera is opened using OpenCV and the video stream is processed. The video is then transformed into frames where image segmentation is used to separate the background. After that the image is compared with the trained model which predicts the label. At last, the label is display to the user in the form of text.

4.7 State Chart Diagram

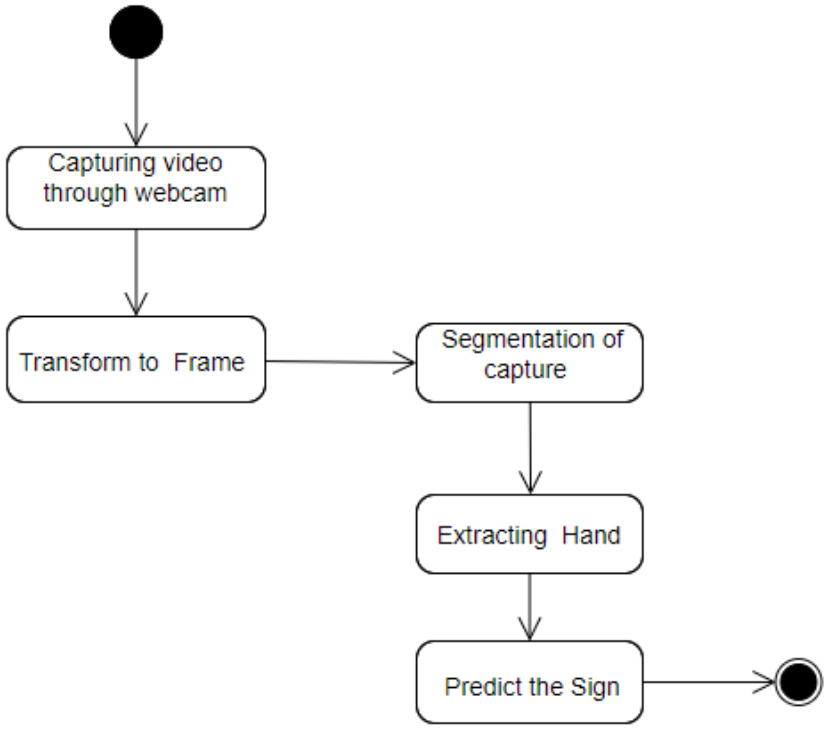


Figure 4.8: State Chart Diagram of Nepali Sign Language Recognition

The above shows the state chart diagram of our system.There are five main state in our system. First the video is captured using a webcam which is transformed as frame.Then segmentation of hand is done followed by extraction of hand. Lastly,the prediction of sign is done.

4.8 Class Diagram

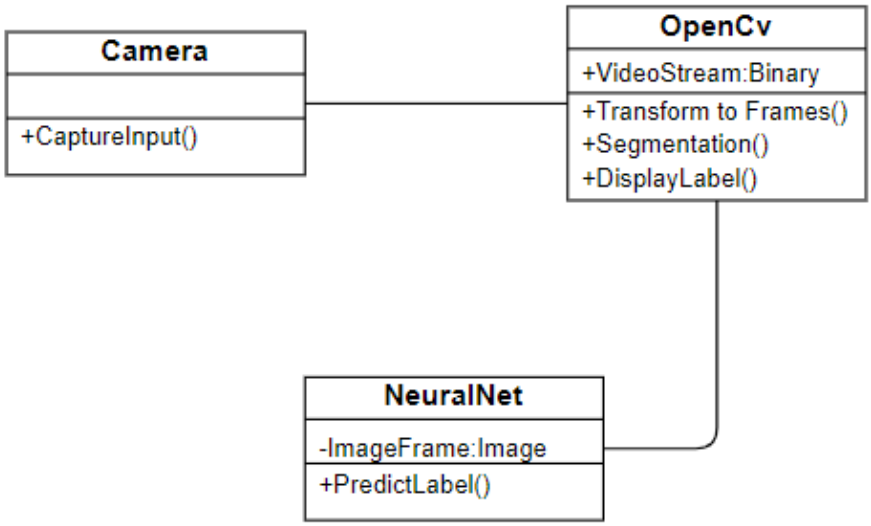


Figure 4.9: Class Diagram of Nepali Sign Language Recognition

The above shows the class diagram of our system.There are three main component which has relation with each other. It is the the blueprints of our system. The camera is used to capture the input which is transform into frames with the help of OpenCV. The input is segmented and the feature is compared with our trained model trained by CNN. After comparing it, the label is predict which is the output.

4.9 Software Development Model

Figure 4.10: Iterative Model

Iterative model is a software development life cycle where initial planning is done after gathering requirements which the datasets which we have prepared by ourselves following the defined nepali sign language. We trained the dataset using CNN model and find out its accuracy. We had tried different models and choose the one which give more predictions.Then we implement the model, test it and evaluate it performing the sign recognition. We had to perform the process iteratively to get the perfect model.

4.10 Techniques

4.10.1 Dataset Preparation

Our proposed system is of Nepali Sign Language(NSL) which dataset was not available so we created the datasets ourselves. For creating the dataset we get the live webcam feed using OpenCV and create an ROI(Region Of Interest) that is nothing but the part of the frame where we want to detect the hand in for the gestures. To removed the background we compute the background's cumulative weighted average in order to distinguish it from the foreground by subtracting it from the frames that have a distinguishable foreground object in front of the background. To achieve this, we compute the accumulated weight for a subset of frames (assume, 60 frames), then we compute the accumulated average for the backdrop.In order to locate any object that covers the background, we deduct the background's cumulative average from each frame we read after 60 frames. we have created dataset of 36 class of different Nepali sign language each class containing 1000 images data.

4.10.2 Image Preprocessing

a. Conversion of RGB Images into GrayScale

   The RGB images contains a lots of color pararmeter as defined by its name Red-green-blue which needed to be reduced for our model. Converting the RGB images into grayscale will help to reduce color complexity and makes its easier for learning our model. It also helps in reduction of noise contains in the images. We uses OpenCV and numpy library for the conversion of RGB images into gray scale.

Figure 4.11: Snapshot of RGB to Gray-Scale Image

b. Blurring of Gray-Scale Images

The dataset that we have created were clicked in the different conditions, some of them were clicked in the low light,and the resulting images has lot of noise, so we used Gaussian Blur which is the way to apply a low-pass filter to mute the noise and for smoothing the images. Gaussian blurring algorithm will scan over each pixel of the image and recalculate the pixel value based on the pixel values that surround it. The area that is scanned around each pixel is called the kernel. A larger kernel scans a larger number of pixels that surround the center pixel.

Gaussian blurring doesn't weigh each pixel equally, however. The closer a pixel is to the center, the greater it affects the weighted average used to calculate the new center pixel value. The image below demonstrates this function. This method assumes pixels closest to the center pixel will be closest to the true value of that pixel, so they will influence the averaged value of the center pixel greater than pixels further away. The Gaussian blur uses a Gaussian function (which also expresses the normal distribution in statistics) for calculating the transformation to apply to each pixel in the image.

Figure 4.12: Snapshot of Blurring of Gray-Scale Image

c. Thresholding the Image

The basic Thresholding technique is Binary Thresholding. For every pixel,

the same threshold value is applied. If the pixel value is smaller than the threshold, it is set to 0, otherwise, it is set to a maximum value(255). Using cv2, we use the threshold function for each frame and identify the contours. Utilizing the function segment, find Contours returns the max contours (the object's outermost contours). We can tell if there is a hand in the ROI by looking at the contours to see if any foreground objects are being picked up there. For the letter we are detecting it for, we begin to save the picture of the ROI in the train and test sets, respectively, when contours are recognized (or a hand is present in the ROI).

dst(x, y) = 0, if threshold >= src(x, y)

dst(x, y) = max value, if threshold < src(x, y)

Figure 4.13: Snapshot of Threshold Image

4.10.3 Splitting of Dataset

Our dataset contains 36,000 images having 1000 images in each 36 different classes of Nepali Sign Languages. For training our model ,we have split each class dataset into 80:20 ratio as train and test. The train sample contains 28,800 and test samples contains 7200.

4.10.4 Training of Convolutional Neural Network model

Our initial dataset contained static photos, so we used a CNN model to categorize them. Our first goal in creating the neural network was to specify the input layer.

By turning each image into a string of integers, we transform the data into a machine-readable format. Once the input layer has been constructed, the neural network's hidden layers will process it. The figure below depicts the architecture of our neural network. The weighted sum of the input values is distributed among the nodes that make up the first hidden layer. The inputs is then passed through an activation function named as rectified linear unit, or ReLU. The ReLU will produce 0 when the input is negative, but will not alter the input otherwise. The ReLU's outputs will be used as inputs for the network's next hidden layer.This model is a sequential neural network with several layers:

1.The first layer is a 2D convolutional layer with 32 filters and a 3x3 kernel size, followed by a max pooling layer with a pool size of 2x2.

2.The second layer is another 2D convolutional layer with 64 filters and a 3x3 kernel size, followed by another max pooling layer with a pool size of 2x2.

3.The third layer is a 2D convolutional layer with 128 filters and a 3x3 kernel size, followed by another max pooling layer with a pool size of 2x2.

4.The fourth layer is a flatten layer to convert the output of the previous layer into a 1D tensor.

5.The fifth layer is a dense layer with 64 units and ReLU activation function.

6.The sixth layer is another dense layer with 128 units and ReLU activation function, followed by a dropout layer with a dropout rate of 0.5 to prevent overfitting.

7.The seventh layer is another dense layer with 128 units and ReLU activation function, followed by another dropout layer with a dropout rate of 0.5 to prevent overfitting.

8.The final layer is a dense layer with 36 units (corresponding to the number of output classes) and softmax activation function to output the probability distribution over the classes.

The model has a total of 417,700 trainable parameters and no non-trainable parameters.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 62, 62, 32)        896

 max_pooling2d (MaxPooling2D  (None, 31, 31, 32)       0
 )

 conv2d_1 (Conv2D)           (None, 31, 31, 64)        18496

 max_pooling2d_1 (MaxPooling  (None, 15, 15, 64)       0
 2D)

 conv2d_2 (Conv2D)           (None, 13, 13, 128)       73856

 max_pooling2d_2 (MaxPooling  (None, 6, 6, 128)        0
 2D)

 flatten (Flatten)           (None, 4608)              0

 dense (Dense)               (None, 64)                294976

 dense_1 (Dense)             (None, 128)               8320

 dropout (Dropout)           (None, 128)               0

 dense_2 (Dense)             (None, 128)               16512

 dropout_1 (Dropout)         (None, 128)               0

 dense_3 (Dense)             (None, 36)                4644

=================================================================
Total params: 417,700
Trainable params: 417,700
Non-trainable params: 0
_________________________________________________________________
```

Figure 4.14: Layers of CNN

CHAPTER 5

RESULT AND ANALYSIS

5.1 Result

5.1.1 Output and Analysis

In this project, initially we created our own dataset through OpenCV. The dataset was created capturing various gestures from the webcam and the entire project was based on image classification. The model received a high accuracy score of 98.64% with 4.76% loss on training set. Similarly, it achieved 98.29% accuracy in validation results.

No.of Epoch	No.of Train Samples	No.of Validation Sample	Training phase Duration	Training Phase Accuracy	Validation Phase Duration	Validation Phase Accuracy
10	28,800	7,200	5hrs 26min	98.64%	29min	98.29%

Table 5.1: Result Table

Epoch	loss	accuracy	val_loss	val_acc
1	3.5133	0.0469	3.2026	0.1259
2	2.997	0.1681	2.5891	0.27
3	2.3983	0.3288	1.8133	0.4834
4	1.6849	0.5261	1.1085	0.6728
5	1.0467	0.6906	0.5702	0.8261
6	0.5555	0.8293	0.1467	0.9559
7	0.2285	0.9318	0.0546	0.9856
8	0.1207	0.9646	0.0249	0.9833
9	0.066	0.9807	0.0124	0.9831
10	0.0476	0.9864	0.0089	0.9829

Table 5.2: Training History

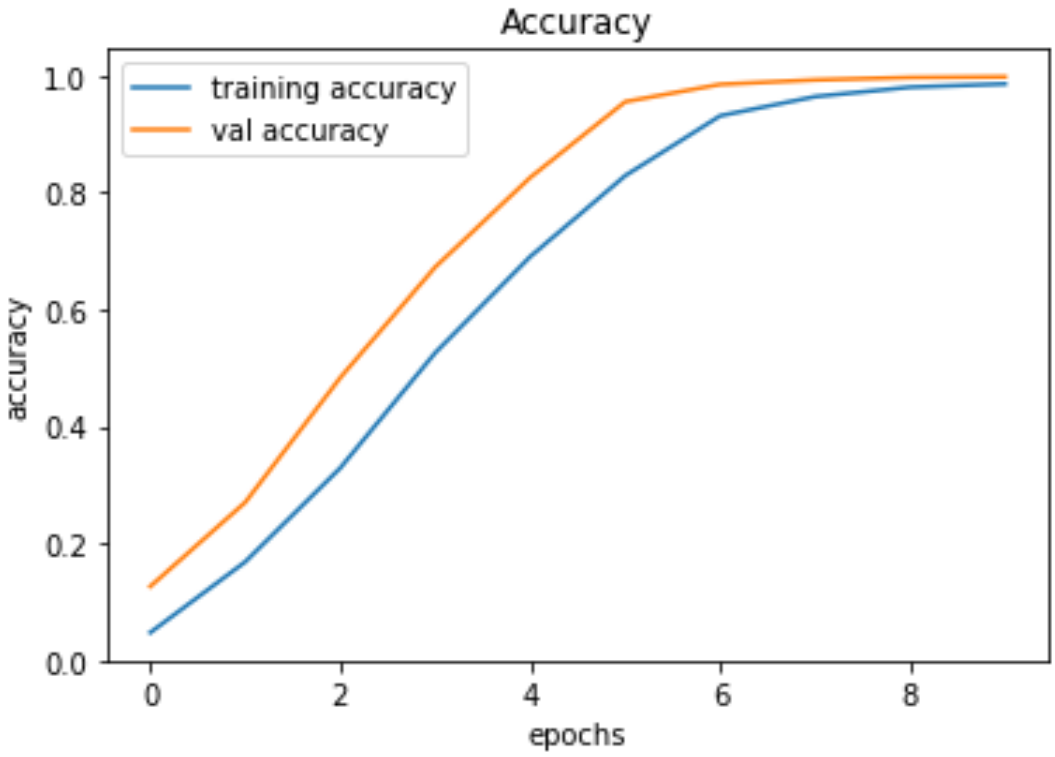


Figure 5.1: Accuracy Graph

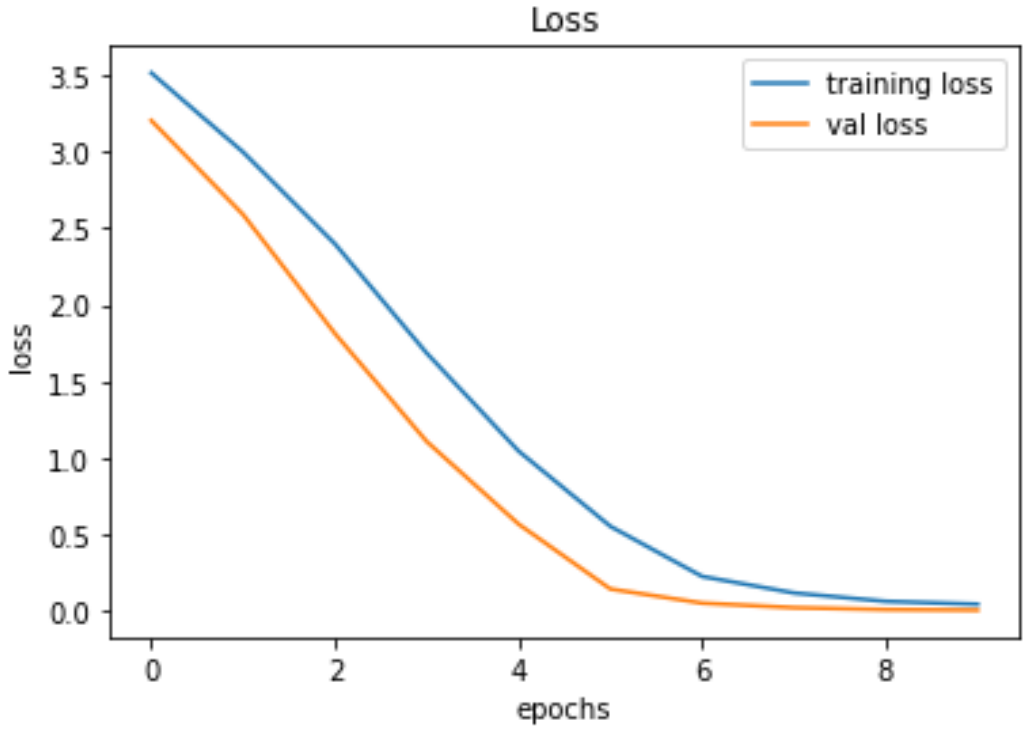


Figure 5.2: Loss Graph

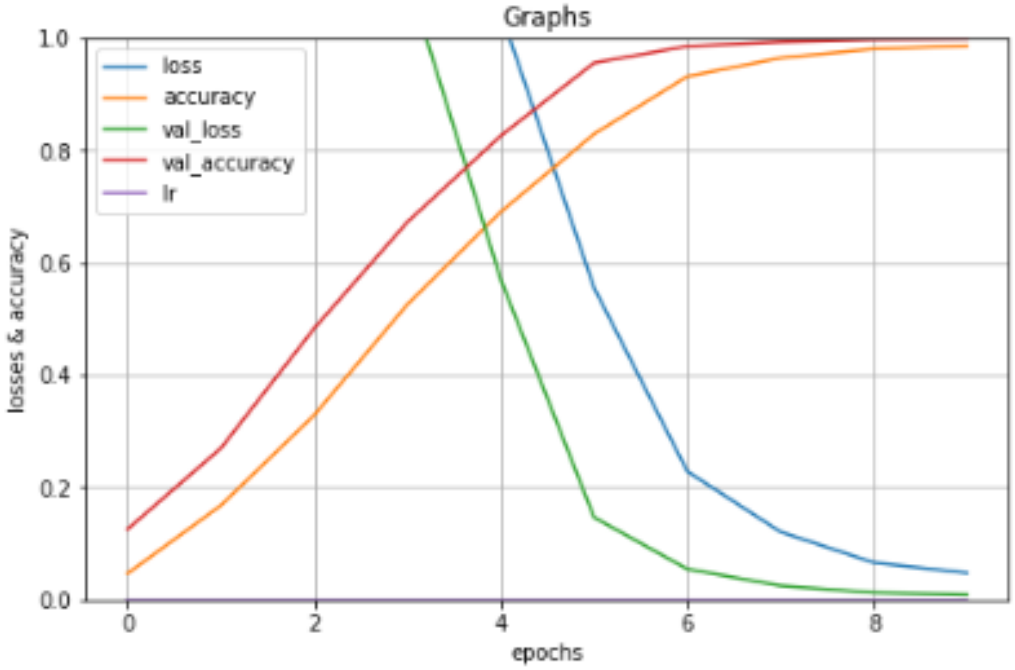


Figure 5.3: Loss VS Accuracy Graph

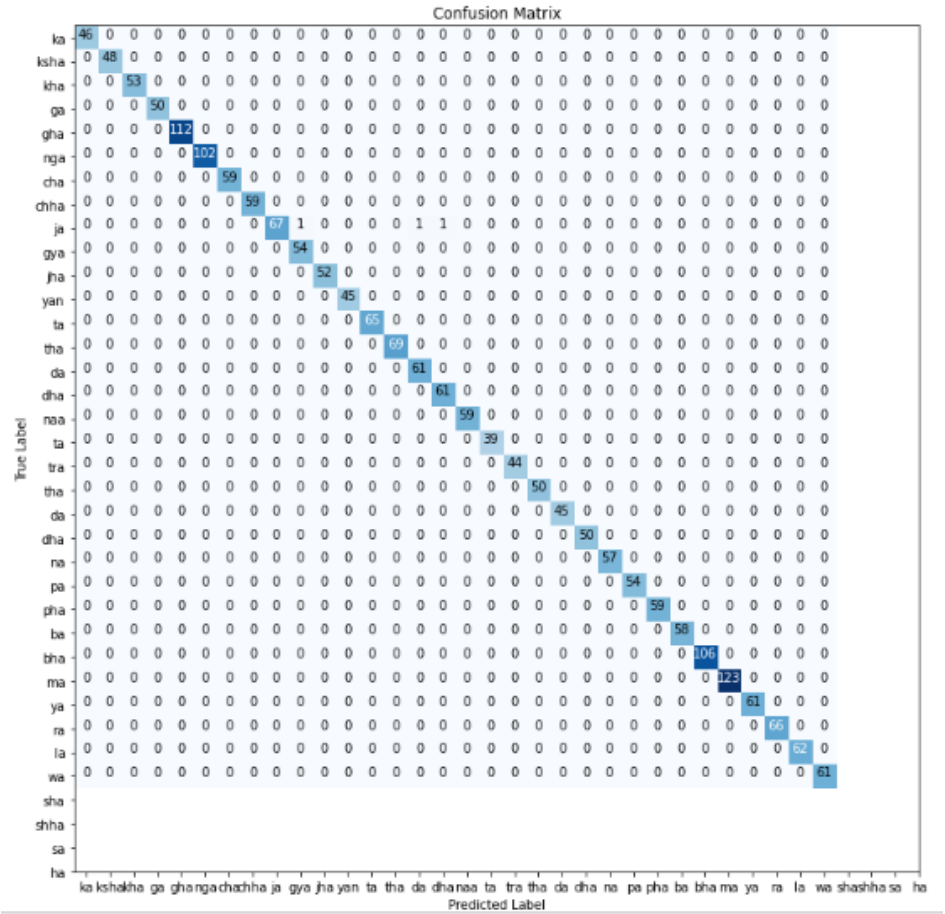


Figure 5.4: Confusion Matrix of Sign Language Detection

The evaluation metrics of our system are:

1.Precision:0.987

2.Recall:0.986

3.F1 Score:0.989

4.Accuracy:0.9852

CHAPTER 6

EPILOGUE

6.1 Conclusion

Hence, the developed Nepali sign language recognition system using CNN successfully recognized 36 different sign hand gestures and achieved a high accuracy score of 98.64% The output is provided in a text format, making it accessible and user-friendly.Overall, the project's success indicates promising prospects for aiding communication accessibility for the hearing-impaired community.

6.2 Future Enhancement

- Improve background Subtraction make the system more accurate.
- Optimize CNN model to reduce the response time and develop a better GUI.

REFERENCES

[1] Ear Aid Nepal. Patient information, n.d. Accessed on: March 5, 2023.

[2] Ying Wu and Thomas S Huang. Nonstationary color tracking for vision-based human-computer interaction. *IEEE transactions on neural networks*, 13(4):948–960, 2002.

[3] Ross Cutler and Matthew Turk. View-based interpretation of real-time optical flow for gesture recognition. In *Proceedings Third IEEE International Conference on Automatic Face and Gesture Recognition*, pages 416–421. IEEE, 1998.

[4] Raymond Lockton and Andrew W Fitzgibbon. Real-time gesture recognition using deterministic boosting. In *BMVC*, volume 2002, pages 1–10, 2002.

[5] Sebastien Marcel, Olivier Bernier, J-E Viallet, and Daniel Collobert. Hand gesture recognition using input-output hidden markov models. In *Proceedings Fourth IEEE International Conference on Automatic Face and Gesture Recognition (Cat. No. PR00580)*, pages 456–461. IEEE, 2000.

[6] Kenji Oka, Yoichi Sato, and Hideki Koike. Real-time fingertip tracking and gesture recognition. *IEEE Computer graphics and Applications*, 22(6):64–71, 2002.

[7] Klimis Symeonidis. Hand gesture recognition using neural networks. *Neural Networks*, 13(5.1), 1996.

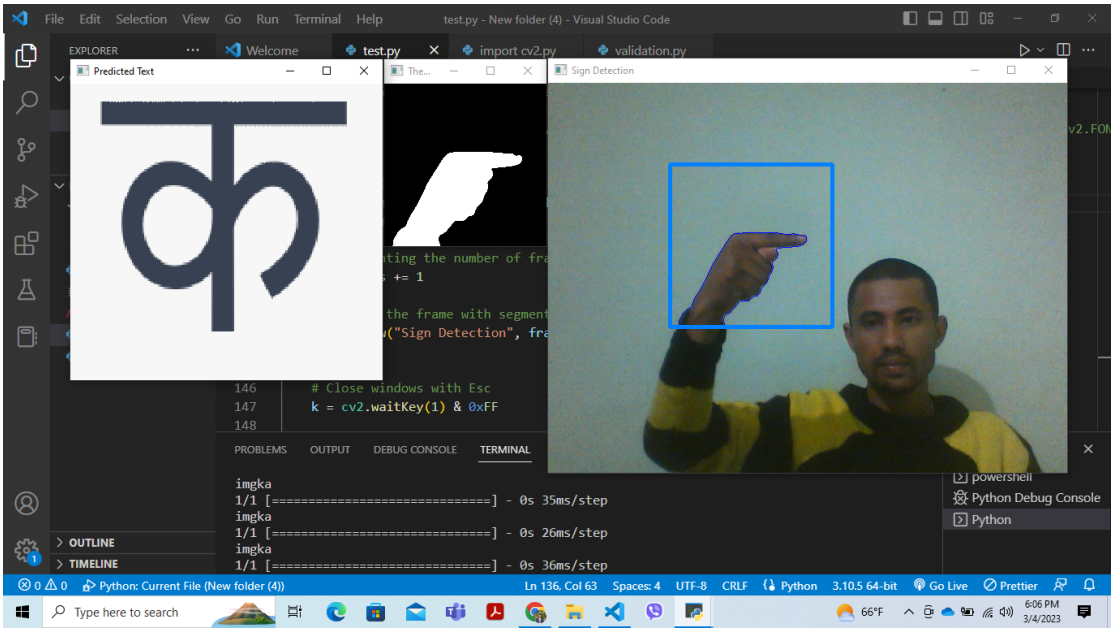
[8] Gregory A Berry. *Small-wall: A multimodal human computer intelligent interaction test bed with applications*. PhD thesis, University of Illinois at Urbana-Champaign, 1998.

[9] Shiqi Yu, Sen Jia, and Chunyan Xu. Convolutional neural networks for hyperspectral image classification. *Neurocomputing*, 219:88–98, 2017.

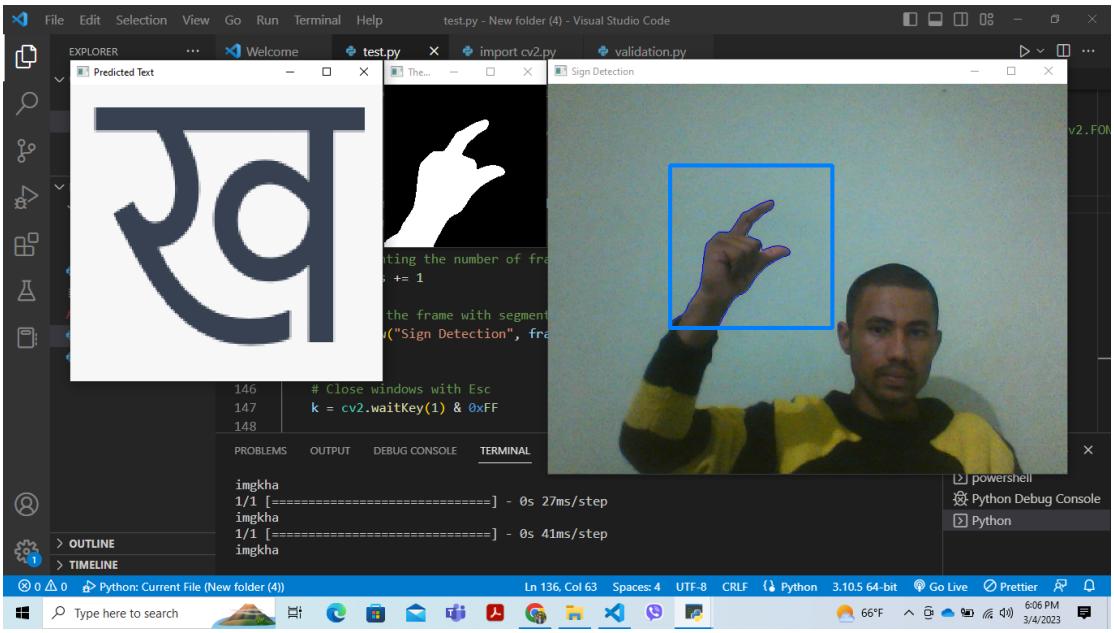
[10] Sumit Saha. A comprehensive guide to convolutional neural networks: The eli5 way, 2018.

APPENDIX

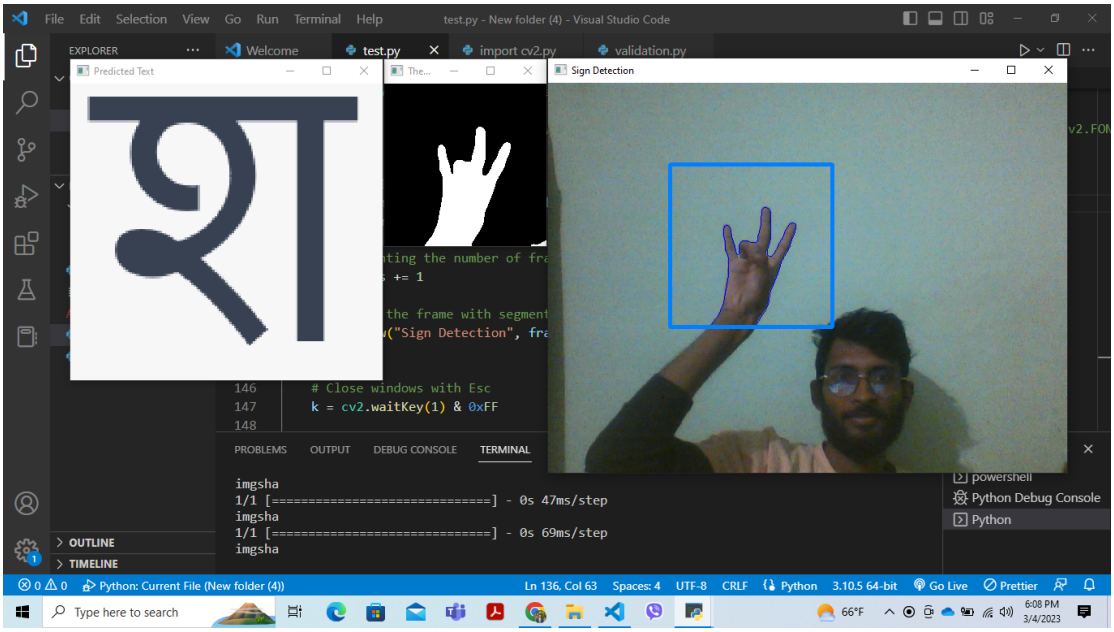
APPENDIX I



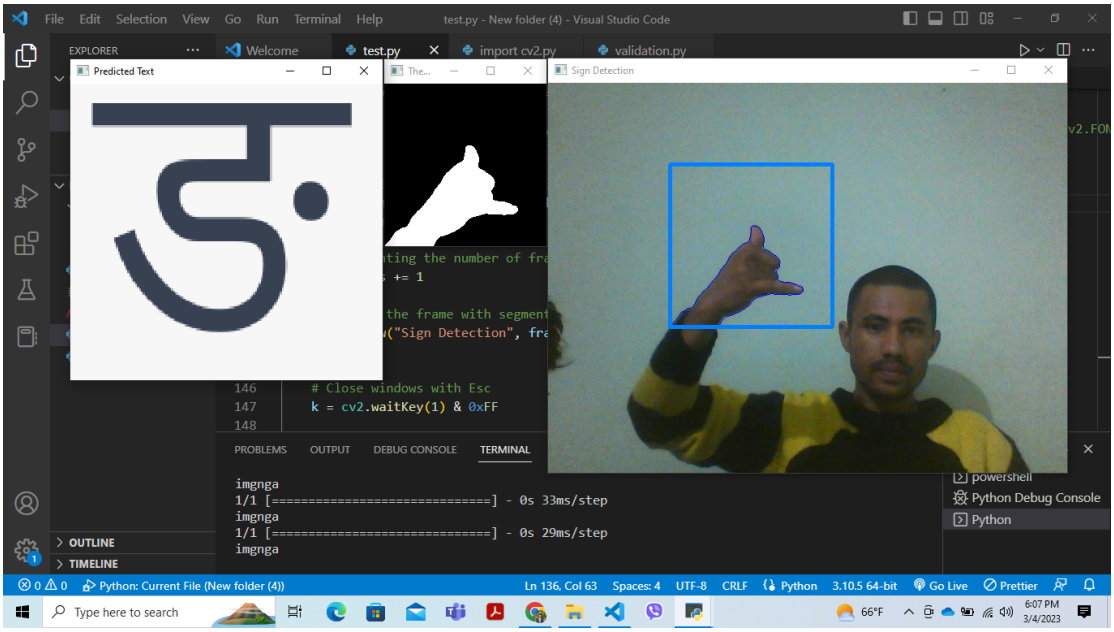
APPENDIX II



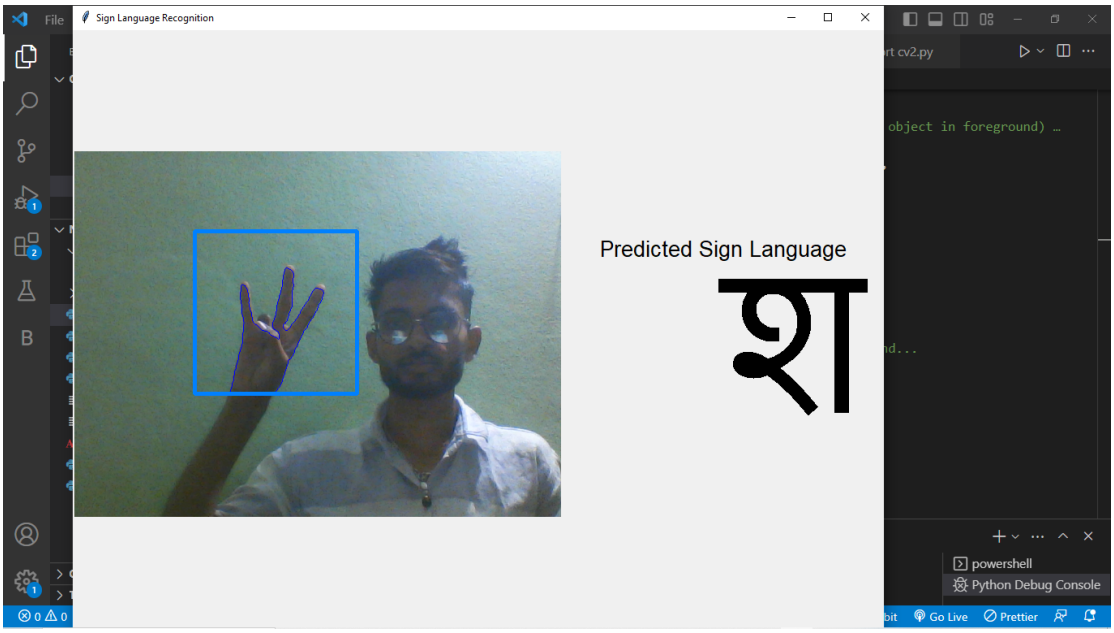
APPENDIX III



APPENDIX IV



APPENDIX V



APPENDIX VI