JSS MAHAVIDYAPEETHA

JSS Science and Technology University

"VISUAL SPEECH RECOGNITION USING DEEP LEARNING"

A technical project report submitted in partial fulfillment of the award of the degree of

BACHELOR OF ENGINEERING

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

BY

Pranav N	01JST18EC069
Prasanna V Bhat	01JST18EC070
Sharat Naik	01JST18EC082
Vaishnavi M	01JST18EC104

Under the guidance of
Prof. Shashidhar R
Assistant Professor
Department of Electronics and Communication Engineering
SJCE, JSS S&TU, Mysuru.

2021-2022
DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

Curiginal

Document Information

Analyzed document	final_draft.pdf (D141945909)
Submitted	2022-07-11 11:30:00
Submitted by	sandeep s
Submitter email	sandeeps@jssstuniv.in
Similarity	12%
Analysis address	sandeeps.jssstu@analysis.urkund.com

Sources included in the report

SA	**JSS Science And Technology University / Final_Report.pdf** Document Final_Report.pdf (D77213519) Submitted by: basappakl64@gmail.com Receiver: basappakl64.jssstu@analysis.urkund.com	15
SA	**JSS Science And Technology University / Report.pdf** Document Report.pdf (D111273907) Submitted by: ise@jssstuniv.in Receiver: ise.jssstu@analysis.urkund.com	1
SA	**JSS Science And Technology University / Phd Thesis_SR.pdf** Document Phd Thesis_SR.pdf (D139044355) Submitted by: sandeeps@jssstuniv.in Receiver: sandeeps.jssstu@analysis.urkund.com	25
SA	**Jamia Hamdard University, Hamdard Nagar / ICIDSSD22_paper_50.pdf** Document ICIDSSD22_paper_50.pdf (D127715500) Submitted by: icidssd22@jamiahamdard.ac.in Receiver: icidssd22.jahau@analysis.urkund.com	1
SA	**PSG College of Technology / Sanjay Balaji.docx** Document Sanjay Balaji.docx (D51040934) Submitted by: UGitg1proj2@psgtech.ac.in Receiver: ugitg1proj2.psgcot@analysis.urkund.com	1
SA	**DP2 Report (2).docx** Document DP2 Report (2).docx (D135528449)	2
W	URL: https://towardsdatascience.com/step-by-step-implementation-3d-convolutional-neural-network-in-keras-12efbdd7b130 Fetched: 2020-04-30 23:32:45	1
W	URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6928880/ Fetched: 2020-03-12 07:44:15	7

DECLARATION

We do hereby declare that the project titled **"Visual Speech Recognition Using Deep Learning"** is carried out by Pranav N, Prasanna V Bhat, Sharat Naik, Vaishnavi M, under the guidance of **Prof.Shashidhar R,Assistant professor , Department of Electronics and Communication Engineering**, JSS Science and Technology University, Mysuru, in partial fulfilment of requirement for the award of Bachelor of Engineering by JSS Science and Technology University, Mysore, during the year 2021-2022.
We also declare that we have not submitted this dissertation to any other university for the award of any degree or diploma courses.

Date:
Place: Mysore

Pranav N
Prasanna V Bhat
Sharat Naik
Vaishnavi M

ACKNOWLEDGEMENT

This project would never have been before your eyes, if it were not for them, their inspiration, patience and support. We express our sincere thanks to **Dr. S B Kivade**, Principal and Dean, JSS Science and Technology University, SJCE, Mysuru for his encouragement and constant support during the entire course.

We owe special thanks to **Dr. U B Mahadevaswamy**, Professor and Head, Department of Electronics and Communication Engineering, JSS Science and Technology University, SJCE, Mysuru, a wonderful person who supported, encouraged and motivated all the students of our batch.

We are grateful to our Guide **Prof. Shashidhar R**, Assistant Professor, Department of Electronics and Communication Engineering, JSS Science and Technology University, SJCE, Mysuru for his valuable guidance, constant encouragement and inspiration towards completion of this project.

We also thank our panel members, Prof. B A Sujathakumari, Associate Professor, Dept of ECE, Prof. Pavitra D R, Assistant Professor, Dept of ECE and Prof. M S Praveen Kumar, Assistant Professor, Dept of ECE, for their support and encouragement towards the completion of the project.

Our hearty thanks to all the other Teaching and Non-Teaching staff for their advice and support throughout the academic study at the department. Last but not the least, we owe a lot to our parents for their constant wishes and encouragement which served as a "Beacon Light" and crowned our efforts with success.

Project Members

Pranav N

Prasanna V Bhat

Sharat Naik

Vaishnavi M

Contents

List of Figures **v**

List of Tables **viii**

1 Introduction **1**

1.1 Preamble 1

1.1.1 Face Detection 2

1.1.2 Lip Reading 3

1.1.3 Deep Learning 3

1.2 Motivation 4

1.3 Problem Statement 4

1.3.1 Objectives 4

1.4 Block Diagram 5

1.5 Organization of the report 6

2 Literature Survey **7**

2.1 Previous research 7

2.2 Summary of literature survey 16

3 Implementation **18**

3.1 Hardware and Software Requirements 18

3.1.1 Hardware Requirements 18

3.1.2 Software Requirements 19

3.2 Dataset Description 21

3.3 Software Implementation 22

3.3.1 Block Diagram 22

3.3.2 Pre-Processing 22
3.3.3 Visual Models 23
3.3.4 Visual Speech Recognition Using CNN 3D Model 24
3.3.5 Visual Speech Recognition Using LSTM Model 34
3.3.6 Visual Speech Recognition Using Hybrid Model 40

4 Results and Discussion 46

4.1 Result for Visual Speech Recognition Using CNN 3D Model 46
4.1.1 CNN 3D Model 47
4.2 Result For Visual Speech Recognition Using LSTM Model 54
4.3 Result for Visual Speech Recognition Using Hybrid Model 61

5 Conlusion and Future Scope 70

5.1 Conclusion 70
5.2 Future Work 70
5.3 Advantages and Limitations 71

Reference 72

List of Figures

1.1 Block Diagram of the project 5

3.1 Gray scale images of Lip localization 23

3.2 Frame Before mouth ROI extraction 24

3.3 Extracting Mouth Region 24

3.4 Convolution 3D dept dimension 26

3.5 Step by step CNN model operation 27

3.6 Model structure of CNN 3D model for English Dataset 28

3.7 Model structure of CNN 3D model for Kannada Dataset 29

3.8 Summary of model structure of CNN 3D for English Dataset 30

3.9 Summary of model structure of CNN 3D for Kannada Dataset 31

3.10 ReLU activation curve 34

3.11 Model structure for LSTM model for English Dataset 35

3.12 Model structure for LSTM model for Kannada Dataset 35

3.13 Summary of model structure of LSTM model for English Dataset. . . . 36

3.14 Summary of model structure of LSTM model for Kannada Dataset. . . 36

3.15 Internal structure of LSTM cell 37

3.16 Structure of Forget Gate 37

3.17 Structure of Input Gate 38

3.18 Structure of Output Gate 39

3.19 Model structure of (CNN 3D + LSTM) hybrid model for English Dataset 40

3.20 Model structure of (CNN 3D + LSTM) hybrid model for Kannada Dataset 41

3.21 Architecture of CNN-LSTM model 41

3.22 **Summary of Model structure of hybrid model (CNN 3D + LSTM) for English Dataset** 42
3.23 **Summary of Model structure of hybrid model (CNN 3D + LSTM) for Kannada Dataset** 43

4.1 **Training English Dataset with CNN 3D model** 48
4.2 **Training Kannada Dataset with CNN 3D model** 48
4.3 **Accuracy plot for CNN 3D model with English Dataset** 48
4.4 **Loss plot for CNN 3D model with English Dataset** 49
4.5 **Accuracy plot for CNN 3D model with kannada Dataset** 49
4.6 **Loss plot for CNN 3D modelwith kannada Dataset** 49
4.7 **Confusion matrix for CNN 3D model with English Dataset** 51
4.8 **Confusion matrix for CNN 3D model with kannada Dataset** 52
4.9 **Classification matrix for CNN 3D model with English Dataset** 54
4.10 **Classification matrix for CNN 3D model with Kannada Dataset** 54
4.11 **Training English Dataset with LSTM model.** 55
4.12 **Training Kannada Dataset with LSTM model.** 55
4.13 **Accuracy plot with English dataset with LSTM model.** 55
4.14 **Loss plot with English dataset with LSTM model.** 56
4.15 **Accuracy plot with Kannada dataset with LSTM model.** 56
4.16 **Loss plot with Kannada dataset with LSTM model.** 56
4.17 **Confusion matrix with LSTM model for English Dataset** 58
4.18 **Confusion matrix for LSTM model for Kannada Dataset** 59
4.19 **Classification matrix with LSTM model for English Dataset** 61
4.20 **Classification matrix with LSTM model for Kannada Dataset** 61
4.21 **Training dataset with hybrid model for English Dataset** 62
4.22 **Training dataset with hybrid model for Kannada Dataset** 62
4.23 **Accuracy plot with hybrid model with English Dataset** 63
4.24 **Loss plot with hybrid model with English Dataset** 63
4.25 **Accuracy plot with hybrid model for Kannada Dataset** 64
4.26 **Loss plot with hybrid model for Kannada Dataset** 64
4.27 **Confusion matrix of the English dataset with hybrid model** 66
4.28 **Confusion matrix of the Kannada dataset with hybrid model** 67

4.29 **Classification matrix of the Kannada dataset with hybrid model** 68
4.30 **Classification matrix of the Kannada dataset with hybrid model** 68

List of Tables

3.1 **Words used to develop Visual Speech Recognition** 22

4.1 **Dataset Description** 47

4.2 **Results of proposed method is compared with existing method for audio visual speech Recognition** 69

ABSTRACT

Communication is the essential part of a human being's life as it enables people to interact with the world and share knowledge with each other. For hearing impairments, it is difficult to communicate without the assistance of lip reading, face expression or signs. Also, it is difficult for them to interpret lip movements without training and hence it is difficult for them to detect the spoken words. To overcome this problem, came up with the Visual Speech Recognition (VSR) systems for lip reading and this makes the hearing impaired to understand speech. Advances in deep learning have attracted a lot of research attention lately in Visual Speech Recognition.Here, we proposed the LSTM & CNN 3D hybrid model for visual speech recognition for MIRACL VC1 Dataset.The process of conversion of the text with no audio when the lip features of the person are extracted for tracking the pattern is formed. Visual speech recognition is a model used to recognize speech automatically. To enhance the accuracy here proposed an idea of integrating two individual models CNN 3D and LSTM as hybrid models. So for the hybrid model, we got the training accuracy of 98%, test accuracy of 85% and validation accuracy of 86% which significantly improved from the individual CNN 3D model for which testing accuracy is 79% and for LSTM model the accuracy was 85% when we used English Dataset. For the Kannada dataset, we got the training accuracy of 72.89%, test accuracy of 61% and validation accuracy of 54.67% for hybrid model. But for individual CNN 3D model the test accuracy obtained is 57.33% and for LSTM model the test accuracy is 49%.

Keywords: VSR, CNN 3D, LSTM, Hybrid Model, Deep Learning, Keras, Python

Acronyms

CNN 3D - Convolutional Neural Network 3d

LSTM - Long Short Term Memory

VSR - Visual Speech Recognition

ANN - Artificial Neural Network

AI - Artificial Intelligence

AAM - Active Appearance Model

WHO - World Health Organization

HMM - Hidden Markov Model

GMM - Gaussian Mixture Modeling

RNN - Recurrent Neural Network

MFCC - Mel-Frequency Cepstrum Coefficient

ASR - Automatic Speech Recognition

AVSR - Audio Visual Speech Recognition

GRU - Gated Recurrent Units

MRNN - Multimodal Recurrent Neural Network

SNR - Signal-to-Noise Ratio

BLSTM - Bidirectional Long Short-Term Memory

PCA - Principal Component Analysis

MTL - Multitask Learning

AV-ASR - Audio Visual Automatic Speech Recognition

CTC - Connectionist Temporal Classification

VAD - Ventricular Assist Device

ResNet - Residual Neural Network

BBC - British Broadcasting Corporation

CER - Character Error Rate

RNN-T - Recurrent Neural Network-Transducer

RNN-LMs - Recurrent Neural Network Language Models

WSJ - Wall Street Journal

AV - Audio-Visual

Chapter 1

Introduction

This chapter gives a brief overview of visual speech recognition and its history. The discrete steps involved in automatic lip reading is also discussed followed by the motivating factors that lead the team to take up this work. The problem is then defined and objectives are set that defines the course of action of the proposed work.

1.1 Preamble

Visual Speech Recognition, a technique which uses image processing capacity in lip reading to help speech recognition to recognize non deterministic speech. The system of speech recognition and lip reading is done separately and systematically, and then the results of each are mixed at the stage of Feature Fusion. Visual articulation is one of the important sources of information in face to face speech prediction. Lip reading is the basis for determining for a large majority of the hearing impaired. When this lip reading is supplemented by electrical signals, it allows easy and efficient understanding of speech in highly trained subjects. The visual information plays a major role in speech perception and prediction.The visual speech recognition integrates lip reading to recognize the speech and convert it to text format.

Visual speech recognition suggests to the detailed feature-based analysis on the lips and its surrounding environment.The various aspects of feature extraction as there is a need for the consideration of the exterior environment and the details that play an important role in speech prediction with the help of the visuals. The VSR comprises complex steps of feature extraction and image processing due to the need of a large diverse database.

Lip- reading is a method to understand speech by observing and interpreting the motion of the lips, face and other social cues. Listening to the voice of a speaker is difficult in noisy environments and Visual Speech Recognition model helps in creating technologies for the same. Although most of the information is carried by the audio signal, the visual signal also carries complementary and redundant information. The visual information, will imporove the speech recognition performance significantly if not affected by the acoustic signal even in the noisy environment.

Visual speech recognition is the process of recognizing the speech by the help of the video without audio. VSR models can be built using machine learning or deep learning. Here we are using hybrid algorithms to obtain highest possible accuracy.

1.1.1 Face Detection

Object detection is one of the computer technologies, which is connected to image processing and computer vision. The primary aim of face detection algorithms is to determine whether there is any face in an image or not.For face recognition, face detection is the essential step. It recognizes the faces detected in the image. Face detection which is also a part of the object detection, is used in several applications like bio-metrics, security, entertainment, personal safety, law enforcement, etc.Face detection is a non-trivial computer vision problem which helps in identifying and localizing the faces in the images.Face detection can be performed using the classical feature-based cascade classifier using the OpenCV library.

Face detection is similar to image recognition in which the image under observation is compared bit by bit. Any face related feature changes in the database will not give a valid comparison. Face recognition has an important role in any sort of face related image processing applications. A lot of work in the field of face detection or recognition are being proposed to make it more advanced and efficient. Lip-area extraction is the most significant part of the process to get a good recognition rate. Many innovations have been made for extracting facial image from the face.The active appearance model (AAM) is a kind of model in which the shape as well as gray-level appearance (the one which shows only shades of gray color with no other colors) may be determined. It is hard to directly identify or recognize lip regions because various other parts like the mustache, eyes, nose, eyebrows, and body are observed in the target image.

For our implementation we have used the dlib library. The dlib library is arguably one of the most utilized packages for face recognition. Dlib is a C++ toolbox for employing machine learning techniques to solve real-world problems. Despite being built in C++, it includes Python bindings executed in Python. The dlib frontal face detector extracts features using Histogram of Oriented Gradients (HOG) and then processes using an SVM. Dlib is used in detecting the face in a frame or image. For detecting different landmarks in an image we will use shape predictor 68 face landmarks model.

1.1.2 Lip Reading

Lip localization is an actively undergoing research topic with a lot of improvements that recovers various difficulties faced in the research. The research of this topic started many years ago. But still the topic is analyzed with various problems and developed with improved accuracy levels for different applications. Lip localization is also known as lip segmentation. Lip localization is the first stage of the Lip reading process. Lip reading is a process where visual information is extracted from the video by watching lip movements of the speaker with or without sound. For visual information extraction reliable mouth movements are required.

The methods used for extracting lips may be semi-automatic or automatic. The semiautomatic methods need to have manual selection of a pixel point to initiate the process in the first image or frame whereas automatic methods will not require manual selection and it will automatically start the process to localize the lips on the face.

1.1.3 Deep Learning

Deep Learning is a subfield of machine learning concerned with algorithms. It is inspired by the structure and function of Artificial Neural Networks(ANN).In addition to scalability, another benefit of deep learning models is their capability to perform automatic feature extraction from raw data which is also called feature learning. The use of these Deep Learning techniques has allowed significant advancement in lip-reading in Visual Speech Recognition.

Deep learning is a type of machine learning and artificial intelligence (AI) that imitates the way humans gain certain types of knowledge. Deep learning is an important element of data science, which includes statistics and predictive modeling. It is extremely beneficial to data scientists who are tasked with collecting, analyzing and interpreting large amounts of data; deep learning makes this process faster and easier.

1.2 Motivation

As per the survey, World Health Organization (WHO) reported that there are approximately 63 million people in India alone who are suffering from Significant Auditory Impairment which is almost 6.3% of Indian Population. Among them, there might be adults who lose their hearing ability after ages due to high exposure to noise or may be due to increasing Noise Pollution.
The key motivation of our work is to assist people with speech impairments. Visual speech recognition would positively impact the life of a large number of patients with speech impairments worldwide. Hence VSR would help them in determining the speech effortlessly. In noisy environments, it is really difficult for listeners or the audience to listen to the Speaker. Hence this Visual Speech Recognition model will help in creating technologies to overcome these problems.

1.3 Problem Statement

- Recognizing the most important visual features while performing lip-reading with DL.
- Developing effective hybrid algorithm in order to improve the accuracy of individual algorithm

1.3.1 Objectives

- To collect appropriate dataset.
- To explore the lip localization method.
- To develop an algorithm for Visual speech recognition using a CNN model.
- To develop an algorithm for Visual Speech Recognition using an LSTM model.

- To develop an algorithm for Visual Speech Recognition using Hybrid model.
- To compare the result with existing work.

1.4 Block Diagram

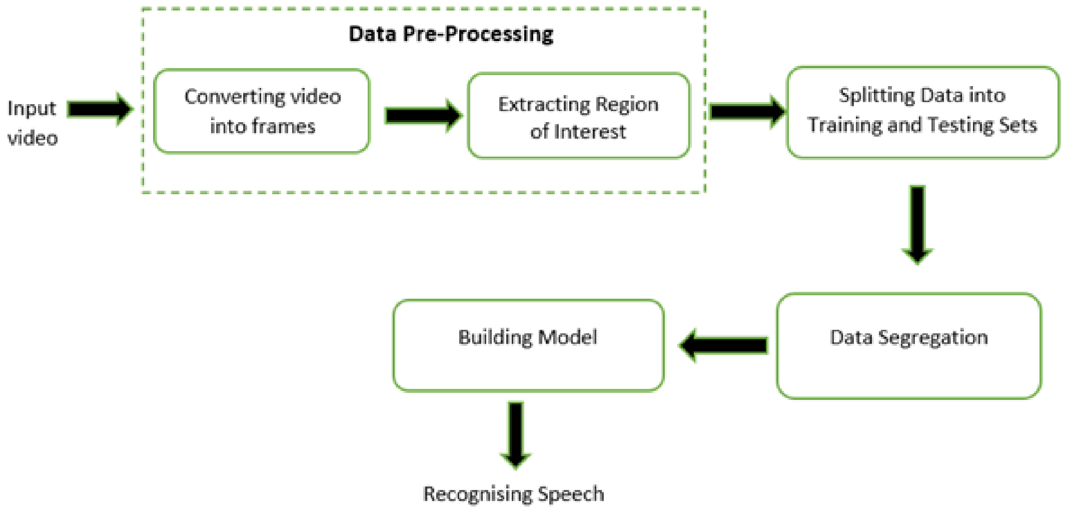

Data Pre-Processing
Input video
Converting video into frames
Extracting Region of Interest
Splitting Data into Training and Testing Sets
Data Segregation
Building Model
Recognising Speech

Figure 1.1: **Block Diagram of the project**

Figure 1.1 shows the block diagram of the proposed model.From the frames of each instance, we will extract the region of interest which is lip in our project. In other words, extracting the region of interest means lip reading. Once the Data Pre-processing is successfully completed, split the data into training, testing and validating sets. Usually more than 70% of the data is used for training and 20% for testing and remaining 10% for validation. This is the main development phase of the project.Then the data is segregated into three subsets as mentioned above. i.e. for the training, testing and validation.

After segregating the data into three subsets, training the processed data into a certain suitable model is done. Here we are using a hybrid model using two models. First the data is trained for two individual models separately and the accuracy is noted for each model. Then to enhance the accuracy, a hybrid model is used by combining the two individual models. Thus all the objectives for the Visual Speech Recognition using deep learning are satisfied.

1.5 Organization of the report

This report is organized in this way.
Chapter 1 gives the overview of the entire project and the motivation to select the project. I also gives the objectives and the bock diagram of the project.
Chapter 2 describes the previous research work on Visual Speech Recognition using different deep learning,machine learning algorithms.After briefly studying the previous works, we have briefly summarized the entire study that helped us while developing Visual Speech Recognition.
Chapter 3 briefly gives the hardware and software requirements to develop Visual Speech Recognition using Deep Learning. In this chapter, we have described the softwares, IDEs, and libraries used to develop Visual Speech Recognition.
Chapter 4 consists of all the system architecture and technical implementations for developing Visual Speech Recognition. Various steps used in the development of the Project.
Chapter 5 explains the Intermediate and final results of the proposed model for the development of the project.The results are explained with mathematical equations and results after each step are explained with convenient photos.
Chapter 6 is about the conclusion of the project with future scope, applications, advantages and disadvantages.

Chapter 2

Literature Survey

In this Chapter, a short description of all the research papers we referred to while starting this project is given. The papers we referred to and researched were really helpful in each stage of the project.To thoroughly understand the previous research on Visual Speech Recognition, we collected as many research papers as possible. So we went on through all those papers, studied and briefly wrote the abstract of all the papers here.

2.1 Previous research

Eric Tatulli et al.[11] gave a good description on “Feature Extraction Using Multimodal Convolutional Neural Networks for Visual Speech Recognition” [1] where they addressed the problem of continuous speech recognition from visual information only, without exploiting any audio signal. Their approach combined a video camera and an ultrasound imaging system for monitoring simultaneously the speaker’s lips and the movement of the tongue. Later, they investigated the use of convolutional neural networks (CNN) to extract visual features directly from the raw ultrasound and video images. The article proposed different architectures among which a multimodal CNN processing jointly the two visual modalities. Combined with an HMM-GMM decoder, the CNN-based approach outperformed their previous baseline based on Principal Component Analysis. Importantly, the recognition accuracy was only 4% lower than the one obtained when decoding the audio signal, which made it a good candidate for a practical visual speech recognition system.
Kai-Xian Lau et al.[14] proposed Audio-Visual Speech Recognition System Using Recurrent Neural Network.In this, at starting before visual information came into picture they

used mainly MFCC(Mel-frequency cepstrum coefficient)even though they said that hmm is more effective in ASR, but its dataset contains enormous amount of data.So RNN has been chosen.The RNN-based AVSR system enhances the accuracy of speech recognition performance in a noisy setting as it takes extra sensory modal to counterpart corrupted audio or undesired background sound. This proposed RNN-based AVSR system uses MFCC for speech feature extraction.In this paper audio information is extracted from unidirectional multimodal LSTM and image information is taken from ANN. And for integration of both they used RNN. Number of hidden Neutrons in audio feature extraction -100,visual feature extraction mechanism -100,Integration mechanism -10 ,Using any of RNN, LSTM, GRU loss is same and accuracy is about 90%.MFCC - 95.4% accuracy.

Greg Pottieet al.[9] proposed comparative analysis of the hidden markov model and lstm:A simulative approach.In this paper,they have done a comparative analysis of e supervised and unsupervised hidden Markov model to LSTM in terms of the amount of data needed for training, complexity, and forecasting accuracy. The final result of this paper says that even an unsupervised hidden Markov model can outperform LSTM when a massive amount of labeled data is not available.The HMM is the extension of the Markov process in which the observations are a probabilistic function of the states.In an HMM, states are considered as hidden and should be inferred by the sequence of observations.A Gaussian distribution or mixture of Gaussian distributions are typically used for modeling the data.RNN networks are mainly designed for the sequence prediction problem.

Brendan Shillingfordet al.[8] proposed large scale visual speech recognition.This paper presents a solution to open-vocabulary visual speech recognition.To achieve this, they have constructed the largest existing visual speech recognition dataset, consisting of pairs of text and video clips of faces speaking (3,886 hours of video). And then they designed and trained an integrated lipreading system, consisting of a video processing pipeline that maps raw video to stable videos of lips and sequences of phonemes, a scalable deep neural network that maps the lip videos to sequences of phoneme distributions, and a production-level speech decoder that outputs sequences of words.

Weijiang Feng et al. [13] addressed on “Audio Visual Speech Recognition with Multimodal Recurrent Neural Networks” According to their study, although existing deep learning models succeed to incorporate visual information into speech recognition, none of them simultaneously considers sequential characteristics of both audio and visual modalities. To overcome this deficiency, they proposed a multimodal recurrent neural network (multimodal RNN) model to take into account the sequential characteristics of both audio and visual modalities for AVSR. Multimodal RNN includes three components, i.e., audio part, visual part, and fusion part, where the audio part and visual part capture the sequential characteristics of audio and visual modalities, respectively, and the fusion part combines the outputs of both modalities.

Pan Zhou et al. [15], published an article on "Modality Attention for End-to-end Audiovisual Speech Recognition" Audio-visual speech recognition (AVSR) system is thought to be one of the most promising solutions for robust speech recognition, especially in noisy environments. In this paper, a novel multimodal attention based method for audio-visual speech recognition which could automatically learn the fused representation from both modalities based on their importance is proposed. The method is realized using state-of-theart sequence-to-sequence (Seq2seq) architectures. Experimental results show that relative improvements from 2% up to 36% over the auditory modality alone are obtained depending on the different signal-to-noise-ratio (SNR).

Zuwei Li et al. [6] addressed on "End-to-end visual speech recognition with LSTMS " Traditional visual speech recognition systems consist of two stages, feature extraction and classification. In this work, an end-to-end visual speech recognition system based on LongShort Memory (LSTM) networks is used. This is the first model which simultaneously learns to extract features directly from the pixels and perform classification and also achieves stateof-the-art performance in visual speech classification. The model consists of two streams which extract features directly from the mouth and different images, respectively. The temporal dynamics in each stream are modeled by an LSTM and the fusion of the two streams takes place via a Bidirectional LSTM (BLSTM). An absolute improvement of 9.7% over the baseline is reported on the OuluVS2 database, and 1.5% on the CUAVE database when compared with other methods which use a similar visual front-end.

Marina Zimmermann et al. [16] published an article on "Visual Speech Recognition Using PCA Networks and LSTMs in a Tandem GMM-HMM System" In this work, principal component analysis is applied to the image patches extracted from the video data to learn the weights of a two-stage convolutional network. The results show that the proposed method has outperformed the baseline techniques applied to the OuluVS2 audiovisual database for phrase recognition with the frontal view cross-validation and testing sentence correctness reaching 79% and 73%, respectively, as compared to the baseline of 74% on cross-validation.

Fei Tao et al. [10], addressed on "End-to-End Audiovisual Speech Recognition System With Multitask Learning" This paper proposes a novel end-to-end, multitask learning (MTL), audiovisual ASR (AV-ASR) system. They obtained a robust and accurate audiovisual system that generalizes across conditions. By detecting segments with speech activity, the AV-ASR performance improves as its connectionist temporal classification (CTC) loss function can leverage the AV-VAD alignment information, the end-to-end system learns from the raw audiovisual inputs a discriminative high-level representation for both speech tasks, providing the flexibility to mine information directly from the data.

Navin Kumar Mudaliar et al. [4] proposed a deep learning model on Visual speech recognition.The proposed visual speech recognition approach has used the concept of deep learning to perform word-level classification. ResNet architecture is used with 3D convolution layers as the encoder and Gated Recurrent Units (GRU) as the decoder. The whole video sequence was used as an input in this approach. The results of the proposed approach are satisfactory. It achieves 90% accuracy on the BBC data set and 88% on the custom video data set. The proposed approach is limited to word-level only and can easily be extended to short phrases or sentences. However the model doesn't perform well with a video containing subjects with facial hair. Thus, more exploration can be done by adding videos to the dataset containing people with facial hair.

Abhinav Thanda et al.[12] proposed a model on Audio Visual Speech Recognition using Deep Recurrent Neural Networks.In this work, the authors proposed a training algorithm for an audiovisual automatic speech recognition (AV-ASR) system using deep recurrent neural network (RNN).First, they trained a deep RNN acoustic model with a Connectionist Temporal Classification (CTC) objective function. The frame labels obtained from the acoustic model are then used to perform a non-linear dimensionality reduction of the visual features using a deep bottleneck network. Audio and visual features are fused and used to train a fusion RNN. The results show that presence of visual modality gives significant improvement in character error rate (CER) at various levels of noise even when the model is trained without noisy data.They also provide a comparison of two fusion methods: feature fusion and decision fusion. The model achieved an overall efficiency of about 90%.

Takaki Makino et al.[1] proposed a model on Audio-Visual Speech Recognition using Recurrent Neural Network Transducers.This work presents a large-scale audio-visual speech recognition system based on a recurrent neural network transducer (RNN-T) architecture. To support the development of such a system,they built a large audio-visual (A/V) dataset of segmented utterances extracted from YouTube public videos, leading to 31k hours of audiovisual training content. The performance of an audio-only, visual-only, and audio-visual system are compared on two large-vocabulary test sets: a set of utterance segments from public YouTube videos called YTDEV18 and the publicly available LRS3-TED set.

Stavros Petridis et al. [7] addressed on "End-to-End Visual Speech Recognition with LSTMS", they present an end-to-end visual speech recognition system based on Long-Short Memory (LSTM) networks. This is the model which simultaneously learns to extract features directly from the pixels and perform classification and also achieves state-of-the-art performance in visual speech classification. The model consists of two streams which extract features directly from the mouth and different images, respectively. The temporal dynamics in each stream are modeled by an LSTM and the fusion of the two streams takes place via a Bidirectional LSTM (BLSTM). An absolute improvement of 9.7% over the baseline is reported on the OuluVS2 database, and 1.5% on the CUAVE database when compared with other methods which use a similar visual front-end.

Yeh-Huann Goh et al.[**14**] proposed an Audio-Visual Speech Recognition System Using Recurrent Neural Network.The proposed AVSR model consists of three components: audio features extraction mechanism, visual features extraction mechanism and audio and visual features integration mechanism. The features integration mechanism combines the output features from both audio and visual extraction mechanisms to generate final classification results. In this research, the audio features mechanism is modeled by Mel-frequency Cepstrum Coefficient (MFCC) and further processed by RNN system, whereas the visual features mechanism is modeled by Haar-Cascade Detection with OpenCV and again, it is further processed by RNN system. Then, both of these extracted features were integrated by a multimodal RNN-based features-integration mechanism. On average, the final speech recognition rate is 89% across different levels of SNR.

Takaaki Hori et al.[3] proposed an end to end speech recognition system using RNN. This paper investigates the impact of word-based RNN language models (RNN-LMs) on the performance of end-to-end automatic speech recognition (ASR). The multi-level approach achieves significant error reduction in the Wall Street Journal (WSJ) task; two different

LMs need to be trained and used for decoding, which increases the computational cost and memory usage. In this paper, authors further propose a novel word based RNN-LM, which allows us to decode with only the word based LM, where it provides look-ahead word probabilities to predict next characters instead of the character-based LM, leading to competitive accuracy with less computation compared to the multi-level LM. We demonstrate the efficacy of the word-based RNN-LMs using a larger corpus, LibriSpeech, in addition to the WSJ we used in the prior work. Furthermore, we show that the proposed model achieves 5.1% WER for the WSJ Eval 92 test set when the vocabulary size is increased, which is the best WERE reported for end-to-end ASR systems on this benchmark.

Aditya Amberkaret al.[**16**] in 2018 proposed a Speech Recognition using Recurrent Neural Networks. This paper presents a study of Recurrent Neural Networks and their performance and the use of RNN by famous Speech to Text conversion engines.

RongfengSuet al.[**17**] proposed a Multimodal Learning Using 3D Audio-Visual Data for Audio-Visual Speech Recognition.Recently, various audio-visual speech recognition (AVSR) systems have been developed by using multimodal learning techniques. One key issue is that most of them are based on 2D audio-visual (AV) corpus with the lower video sampling rate. To address this issue, a 3D AV data set with a higher video sampling rate (up to 100 Hz) is introduced to be used in this paper. Another issue is the requirement of both auditory and visual modalities during the system testing. To address this issue, a visual feature generation based bimodal convolutional neural network (CNN) framework is proposed to build an AVSR system with wider application. In this framework, a long short-term memory recurrent neural network (LSTM-RNN) is used to generate the visual modality from the auditory modality, while CNNs are used to integrate these two modalities.

Sonali et al.[**18**] proposed the basics of neural network paper.This paper shows the basic characteristics and working of ANN. ANN is an information processing model that is influenced by biological nervous systems such as brain Neural networks is a highly interconnected processing element working in union to solve a specific problem such as pattern recognition or data classification,through a learning process.Almost all neural network has the same infrastructure.In output layer 'lateral inhibition' is used before training initial weights are chosen randomly,then training begins.They also said that Performance of neural networks is at least as good as classical statistical modeling, and better on most problems. The neural networks build models that are more reflective of the structure of the data in significantly less time.

Kuniaki Noda et al. [**19**] published an article on “Audio-Visual Speech Recognition” This study introduces a connectionist-hidden Markov model (HMM) system for noise-robust AVSR. First, a deep denoising autoencoder is utilized for acquiring noise-robust audio features. By preparing the training data for the network with pairs of consecutive multiple steps of deteriorated audio features and the corresponding clean features, the network is trained to output denoise audio features from the corresponding features deteriorated by noise. Second, a convolutional neural network (CNN) is utilized to extract visual features from raw mouth area images. Approximately 65% word recognition rate gain is attained with denoised MFCCs under 10dB signal to noise ratio (SNR) for the audio signal input.

Satoshi Tamur et al. [**20**] published an article on “Audio-Visual Speech Recognition Using Deep Bottleneck Features and High-performance Lip Reading” This paper develops an Audio-Visual Speech Recognition (AVSR) method, by exploring high-performance visual features, applying audio and visual deep bottleneck features to improve AVSR performance, and investigating effectiveness of voice activity detection in a visual modality. Many kinds of visual features are incorporated, subsequently converted into bottleneck features by deep learning technology. By using proposed features, they achieved 73.66% lip reading accuracy.

Yuki Takashima et al. [**21**] published a paper on “Audio-Visual Speech Recognition Using Bimodal-Trained Bottleneck Features for a Person with Severe Hearing Loss” They proposed an audio-visual speech recognition system for a person with an articulation disorder resulting from severe hearing loss. In the case of a person with this type of articulation disorder, the speech style is quite different from those of people without hearing loss that a speaker-independent acoustic model for unimpaired persons is hardly useful for recognizing it. The audio-visual speech recognition system in this paper is for a person with severe hearing loss in noisy environments. Although feature integration is an important factor in multimodal speech recognition, it is difficult to integrate efficiently because those features are different intrinsically. A novel visual feature extraction approach that connects the lip image to audio features efficiently, and the use of convolutive bottleneck networks (CBNs) increases robustness with respect to speech fluctuations caused by hearing loss.

Fayeket al.[**22**] proposed a model on real time speech emotion recognition using the Deep Neural Network.They proposed a real-time SER system based on end-to-end deep learning.

Namely, a Deep Neural Network (DNN) that recognizes emotions from a one second frame of raw speech spectrograms is presented and investigated. This is achievable due to a deep hierarchical architecture, data augmentation, and sensible regularization. Promising results are reported on two databases which are the eNTERFACE database and the Surrey Audio-Visual Expressed Emotion (SAVEE) database. Various DNN architectures were explored. Promising results were reported on two databases, which are the eNTERFACE database and the Surrey Audio-Visual Expressed Emotion (SAVEE) database. The proposed SER system has an accuracy of 60.53% classifying all six emotions in the eNTERFACE database and an accuracy of 59.7% classifying all seven emotions in the SAVEE database.
Andrew Zisserman et al.[**23**] proposed The Conversation: Deep Audio-Visual Speech Enhancement.Main focus of this paper is to isolate speakers from multi-talker simultaneous speech in videos.They have proposed a deep audio-visual speech enhancement network that is able to separate a speaker's voice given lip regions in the corresponding video, by predicting both the magnitude and the phase of the target signal.The performance of the model is evaluated for up to five simultaneous voices, and we demonstrate both strong qualitative and quantitative performance. The trained model is evaluated on unconstrained 'in the wild' environments, and for speakers and languages unseen at training time.Visual features are extracted from the input image frame sequence with a spatio-temporal residual network pre-trained on a word-level lip reading task. The network consists of a 3D convolution layer, followed by a 18-layer ResNet.
Shankar et al.[2] proposed Audio Visual Speech Recognition using Deep Recurrent Neural Networks.In this work, they proposed a training algorithm for an audiovisual automatic speech recognition system using deep recurrent neural network.First, the trained a deep RNN acoustic model with a Connectionist Temporal Classification objective function. Audio and visual features are fused and used to train a fusion RNN. The use of bottleneck features for visual modality helps the model to converge properly during training. Their system was evaluated on GRID corpus. The results showed that presence of visual modality gives significant improvement in character error rate (CER) at various levels of noise even when the model is trained without noisy dataFor the feature extraction,the sampling rate of audio data is converted to 16kHz. The video frame rate is increased to match the rate of audio frames through interpolation. For AV-ASR, the ROI for visual features is the region surrounding the speaker's mouth. Character Error Rate(CER) is obtained from the

decoded and expected label sequences by calculating the Edit distance between them.

Yuki Yamaguchi et al.[**25**] proposed Audio -visual speech recognition using deep learning.This paper focuses not only noiseless data sets,but also data corrupted by noise.At first,to remove the noise a deep denoising autoencoder for noise robust audio features.After preparing the data for model,it gets trained to output denoised audio features from corresponding noisy features. Acquired audio feature sequences are then processed with a conventional hidden Markov model with a Gaussian mixture observation model to conduct an isolated word recognition task. For visual features,CNN is utilized to extract visual features of the mouth area in image. Finally,multi stream HMM integrating the acquired audio and visual HMMs independently trained with the respective features. They used a Japanese audiovisual dataset for the evaluation of the proposed models. In the dataset, speech data from six males were used. In total, 24000 word recordings were prepared.

Oriol Vinyals et al.[**26**] proposed Deep Audio-visual Speech Recognition.In this paper they have mainly compared the two models i,e. one is CTC loss, and the other using a sequence-to-sequence loss.Both models are built on transformer self-attention architecture.They also challenged about that their model will surpasses all the previous works in the same In this they have chosen 2 datasets i.e, LRS2-BBC, consisting of thousands of natural sentences from British television; and LRS3-TED, consisting of hundreds of hours of TED and TEDx talks obtained from YouTube.

Navdeep Jaitlyet al.[**27**] proposed towards end-to-end speech recognition with rnn.this paper represents speech recognition model that directly transcribes audio data with text without requiring an intermediate phonotonicrepresentation.This model is based on combination of deep bidirectional LSTM rnn and the connectionist temporal classification objective function,n. A modification to the objective function is introduced that trains the network to minimise the expectation of an arbitrary transcription loss function. This allows a direct optimisation of the word error rate, even in the absence of a lexicon or language model. The system achieves a word error rate of 27.3% on the Wall Street Journal corpus with no prior linguistic information, 21.9% with only a lexicon of allowed words, and 8.2% with a trigram language model. Combining the network with a baseline system further reduces the error rate to 6.7%.

Pooja et al.[**28**] proposed an application to recognize, interpret and execute speech commands across other applications and programs. The absence of a system where spoken commands to a computer are standardised, or at least confined to a dialect, leads to increased complexity in providing speech support. Hence to combat this problem, VAAC - Voice Assist and Control. The system aims at providing sufficient support to complement actions performed using other input devices and enhancing device accessibility by incorporating it in all computing environments.

Naeem Ahmed et al.[5] proposed the introduction of a new method called Class Specific Maximization of Mutual Information (CSMMI) which is used to learn a compact and discriminative dictionary for each class. CSMMI not only discovers the latent classspecific dictionary items that best discriminates against different actions, but also captures unique dictionary items for a specific class. It not only discovers the latent class-specific dictionary items that best discriminates different actions, but also captures unique dictionary items for a specific class.The concept involving Action and Gesture recognition system can be used in many of the areas such as man and machine interface, gaming technology, control of mechanical systems, remote control systems, recognition of sign language, control through facial gestures and eye tracking.

Sindhu et al. [**30**] proposed speech enhancement using Hidden Markov model (HMM) - based minimum mean square error in Mel-frequency domain is mainly focused and to estimate clean speech waveform from a noisy signal, an inversion from the Mel- frequency domain to the spectral domain is required which introduces distortion artefacts in the spectrum estimation and filtering. To obtain a more accurate hidden Markov model (HMM) of noisy speech using the Vector Taylor Series (VTS) which is used to estimate the mean vectors and covariance matrices of HMM for noisy speech.

2.2 Summary of literature survey

From the previous study it can be seen that various methods for example HMM,GMM,KNN,CNN and LSTM have been applied for visual speech recognition. In paper [1] they achieved overall accuracy of 80.44% using CNN method HMM - GMM decoder. In paper [2] we can see that authors have used RNN-based visual speech recognition methods and they were able to achieve overall accuracy of 90% and among these methods CNN has the least accuracy and RNN method has the best accuracy.

In [7] using end to end VSR, an improvement of 9.7% over the baseline reported on the

OuluVS2 database, and 1.5% on the CUAVE database when compared with other methods which use a similar visual front-end. In [3] they proposed an end to end automated speech model which achieves 5.1% WER for the WSJ Eval 92 test set when the vocabulary size is increased, which is the best WERE reported for end-to-end ASR systems on this benchmark.

By comparing the results obtained in the existing works, we will build a hybrid model of Visual Speech Recognition with more accuracy. We are using CNN LSTM hybrid algorithm to enhance accuracy of recognized speech. The key advantage of using this model is to get more accuracy even in the noisy environment.LSTMs provide us with a large range of parameters such as learning rates, and input and output biases. Hence, no need for fine adjustments.

Chapter 3

Implementation

Chapter 3 mainly consists of all the software and hardware requirements of the working project. All the software that is used in developing Visual Speech Recognition model is included in this chapter.
This chapter also deals with the pipelines and methodologies used for the implementation of Visual Speech Recognition. To implement Visual Speech Recognition, a dataset was created with English words. In English dataset, there are totally 1500 instances of 10 different words ‘Begin’, ‘Choose’, ‘Connection’, ‘Navigation’, ‘Next’, ‘Previous’, ‘Start’, ‘Stop’, ‘Hello’, ‘Web’ with each word having 150 odd instances. There are a total of 15 speakers out of which 10 female speakers and 5 male speakers. Each of them utter each word 10 times.

3.1 Hardware and Software Requirements

3.1.1 Hardware Requirements

As our project is completely software based, the one and only hardware requirement is the personal Computer which provides the environment in which the project can be carried out.
Laptop:
A portable computer. The following are the specifications of the laptop for a smooth and efficient run of programs:-
Processor: Ryzen 5 Acer nitro 5- CPU @ 2.1GHz
RAM: 8GB + 4GB Graphics card

Operating System: Windows or Linux

3.1.2 Software Requirements

To develop the model for Visual Speech Recognition using deep learning, python IDE is used. Here, for developing the project, we worked in a colab.

Python Language

Python is a high-level, interpreted, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation.
Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly procedural), object-oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library.
Guido van Rossum began working on Python in the late 1980s as a successor to the ABC programming language and first released it in 1991 as Python 0.9.0. Python 2.0 was released in 2000 and introduced new features such as list comprehensions, cycle-detecting garbage collection, reference counting, and Unicode support. Python 3.0, released in 2008, was a major revision that is not completely backward-compatible with earlier versions. Python 2 was discontinued with version 2.7.18 in 2020.Python consistently ranks as one of the most popular programming languages.
Python is meant to be an easily readable language. Its formatting is visually uncluttered and often uses English keywords where other languages use punctuation. Unlike many other languages, it does not use curly brackets to delimit blocks, and semicolons after statements are allowed but rarely used. It has fewer syntactic exceptions and special cases than C or Pascal.
Python NumPy: NumPy, which stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays. Using NumPy, mathematical and logical operations on arrays can be performed. NumPy is a Python package. **Python OS:**Python OS module provides the facility to establish the interaction between the user and the operating system. It offers many useful OS functions that are used to perform OS-based tasks and get related information about operating systems.

The OS comes under Python's standard utility modules.

Python imageio: Imageio is a Python library that provides an easy interface to read and write a wide range of image data, including animated images, volumetric data, and scientific formats.

Python Dlib: Dlib is a modern C++ toolkit containing machine learning algorithms and tools for creating complex software in C++ to solve real world problems. It is used in both industry and academia in a wide range of domains including robotics, embedded devices, mobile phones, and large high performance computing environments. Dlib'sopen source licensing allows you to use it in any application, free of charge. Dlib Shape Predictor:Shape predictors, also called landmark predictors, are used to predict key (x, y)-coordinates of a given "shape". The most common, well-known shape predictor is dlib's facial landmark predictor used to localize individual facial structures, including the: Eyes, Eyebrows, Nose. Python OpenCV:OpenCV is a huge open-source library for computer vision, machine learning, and image processing. OpenCV supports a wide variety of programming languages like Python, C++, Java, etc. It can process images and videos to identify objects, faces, or even the handwriting of a human. When it is integrated with various libraries, such as Numpy which is a highly optimized library for numerical operations, then the number of weapons increases in your Arsenal i.e whatever operations one can do in Numpy can be combined with OpenCV.

Python Imutils: Imutils are a series of convenience functions to make basic image processing functions such as translation, rotation, resizing, skeletonization, and displaying Matplotlib images easier with OpenCV.

Python Tuple: Tuples are used to store multiple items in a single variable.Tuple is one of 4 built-in data types in Python used to store collections of data, the other 3 are List, Set, and Dictionary, all with different qualities and usage.A tuple is a collection which is ordered and unchangeable.Tuples are written with round brackets.

Keras

Keras is the deep learning Application programming interface written in Python, running on top of the ML(Machine Learning) platform with TensorFlow. It has been developed with a sheer focus in enabling tremendous and fast experimentation. Keras has been able to go from idea to the expected result as fast as possible. Keras is Simple but definitely not simplistic. It reduces developer a good amount of cognitive load which frees the user to focus on the parts of the problem which have more importance and actually matter. Keras is Flexible. It has the ability to adopt the principle of progressive disclosure of complexity, like simple workflows are made quick and easy, while arbitrarily advanced workflows are made possible with the help of a clear path that builds upon what the user has already learned. Keras is Powerful. It provides the required industry strength, scalability and performance. it is used by prestigious companies and organizations including Waymo, NASA, and YouTube.

Google Colab

Colab is the commonly used abbreviation of the New York City artists' group Collaborative Projects, which was formed after a series of open meetings between artists of various disciplines.Colab members came together as a collective in 1977, first using the name Green Corporation, and initially received a National Endowment for the Arts (NEA) Workshop Grant through Center for New Art Activities, Inc., a small not-for-profit organization formed in 1974. In 1978, Collaborative Projects was incorporated as a not-for-profit of its own. By raising its own sources of funding, Colab was in control of its own exhibitions and cable TV shows.

3.2 Dataset Description

MIRACL-VC1 is a lip-reading dataset including both depth and color images. It can be used for diverse research fields like visual speech recognition, face detection, and biometrics. Fifteen speakers out of which 5 men and 10 women positioned themselves in the frustum of a MS Kinect sensor and uttered ten times a set of ten words and ten phrases (see the table below). Each instance of the dataset consists of a synchronized sequence of color and depth images (640x480 pixels). The MIRACL-VC1 dataset contains a total number of 3000 instances. For our project we used only color images.Table 3.1 shows the Dataset used for the proposed model.

ID	Words	Resolution
1	Begin	640 x 480
2	Choose	640 x 480
3	Connection	640 x 480
4	Navigation	640 x 480
5	Next	640 x 480
6	Previous	640 x 480
7	Start	640 x 480
8	Stop	640 x 480
9	Hello	640 x 480
10	Web	640 x 480

Table 3.1: **Words used to develop Visual Speech Recognition**

3.3 Software Implementation

3.3.1 Block Diagram

Project is completed by conducting three steps. The Block Diagram which consists of all the methodologies included in the three steps is shown in the figure 3.1.

1. Pre-Processing
2. Visual Models
3. Hybridization of Models

First we will get the Region of Interest from the frames and then using Deep Learning algorithms, two individual models are created and then the two models are combined to produce the hybrid model for Visual Speech Recognition.

3.3.2 Pre-Processing

Dataset contains videos for every person speaking every phrase for ten times. Such videos which are recorded at 10 FPS are of large size and the pre-processing begins with conversion of the videos to frame size. For every instance we have around 10-15 frames saved in the same directory structure. Such pre-processed datasets are now commonly available for free to use like the MIRACL-VC1 dataset. Once the frames are obtained, every frame is cropped to extract the ROI (Region of interest) i.e. the mouth region which is done by the facial landmark method which is a function available in the Dlib library. Parallelly the images are converted in gray scale to further compress the dataset. The Shape predictor covers a human face into 68 points from which the range of 48-68 covers the region of mouth.

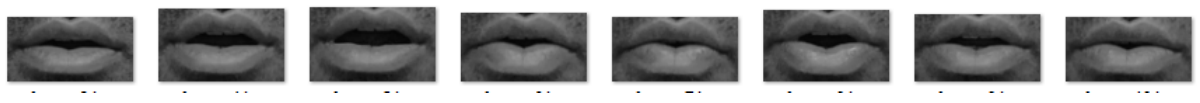


Figure 3.1: **Gray scale images of Lip localization**

3.3.3 Visual Models

From the dataset, the mouth region is extracted using the Face landmark detection algorithm which is available in the dlib library which is a python 3 library in each frame of the video.Then this lip region is converted into gray scale as it is computationally compatible. Then the position of the outer lip coordinates are extracted and the values are used further. The outer lip coordinates are between 48 to 68.

The lip localized value is saved as a python tuple as (x, y, w, h) where (x,y) are the coordinates and w is the width, and h is the height. Then separately the lip coordinates (x,y) are stored in an array using python numpy library. Typically 68 facial landmarks are detected out of which for the lip localization only the landmarks from 48 to 68 are used and other landmarks are masked.

Only the required facial landmarks are detected and the images are cropped and saved using dlib shape predictor. Now the image is loaded, resize of image takes place and then it is converted into grayscale image.

Now the required words are loaded for training the processed data. These words are taken by the folder and if the folder is not present, the folder is created and the directory path is changed and all the data is stored in that particular folder. Then the processed data is used in developing CNN model for Visual Speech recognition and the model is explained below.

Figure 3.2: **Frame Before mouth ROI extraction**

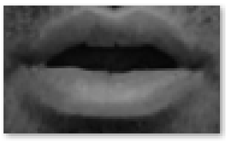


Figure 3.3: **Extracting Mouth Region**

3.3.4 Visual Speech Recognition Using CNN 3D Model

3D convolutions are used when we want to extract features in 3D or establish a relationship between 3D.Essentially, it’s the same as 2D convolutions, but the kernel movement is now 3D, causing a better capture of dependencies within the 3D and a difference in output dimensions post convolution.The kernel of the 3D convolution will move in 3D if the kernel’s depth is lesser than the feature map’s depth.

In deep learning, a convolutional neural network also popularly known as CNN or ConvNet is a class of deep neural networks, most commonly applied to analyze visual imagery. Now when we think of a neural network we think about matrix multiplications but that is not the case with ConvNet. It uses a special technique called Convolution. Now in mathematics convolution is a mathematical operation on two functions that produces a third function that

expresses how the shape of one is modified by the other.
Convolutional neural networks are composed of multiple layers of artificial neurons. Artificial neurons, a rough imitation of their biological counterparts, are mathematical functions that calculate the weighted sum of multiple inputs and output an activation value. When you input an image in a ConvNet, each layer generates several activation functions that are passed onto the next layer.
The first layer usually extracts basic features such as horizontal or diagonal edges. This output is passed on to the next layer which detects more complex features such as corners or combinational edges. As we move deeper into the network it can identify even more complex features such as objects, faces, etc.
Convolution: Convolution is an orderly procedure where two sources of information are intertwined, it's an operation that changes a function into something else. Convolutions have been used for a long time typically in image processing to blur and sharpen images, but also to perform other operations. For e.g. enhance edges and emboss. CNNs enforce a local connectivity pattern between neurons of adjacent layers.
CNNs make use of filters (also known as kernels), to detect what features, such as edges, are present throughout an image. There are four main operations in a CNN:

- Convolution
- Non Linearity (ReLU)
- Pooling or Sub Sampling
- Classification (Fully Connected Layer)

The 3D Convolution Layer A 3D CNN remains regardless of what a CNN that is very much similar to 2D CNN. Except that it differs in few points like non-exhaustive listing.Originally a 2D Convolution Layer is an entry per entry multiplication between the input and the different filters, where filters and inputs are 2D matrices.In a 3D Convolution Layer, the same operations are used. We do these operations on multiple pairs of 2D matrices.
3D convolutions apply a 3D filter to the dataset and the filter moves 3- direction (x, y, z) to calculate the low level feature representations. Their output shape is a 3D volume space such as cube or cuboid. They are helpful in event detection in videos, 3D medical images

etc. They are not limited to 3D space but can also be applied to 2D space inputs such as images.Figure 3.4 shows the convolution 3D dimensions.

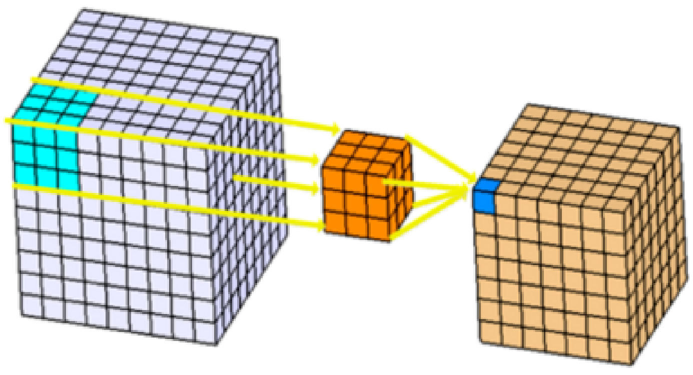


Figure 3.4: **Convolution 3D dept dimension**

Different from 2D-CNN, feature learning is extended to the depth dimension by using 3D convolution kernels. The 3D convolution kernels conduct inner product across the spatial dimension and the depth dimension. Thus, it is suitable for the feature learning of hyper spectral data with rich spectral information. Firstly, R-Hybrid extracts spatial– spectral features using multi-scale 3D convolution kernels, in which padding is used to ensure the consistent dimension of input and output data. The operation results are concatenated in the spectral dimension as the output feature map of the first layer. The next step is to further integrate and abstract the spatial–spectral features with 3 successive convolutional layers. The process of 3D convolution operation is shown in Figure 3.4. The calculation method of the position (x, y, z) of the kth feature in the jth convolutional layer is shown in Formula

$$V_{(j,k)}^{(x,y,z)} = \sum_{m} \sum_{(h=0)}^{(H_i-1)} \sum_{(w=0)}^{(W_j-1)} \sum_{(c=0)}^{(C_j-1)} k_{(j,k,m)}^{(h,w,c)} V_{((j-1),m)}^{((x+h),(y+w),(z+c))} + b_{(j,k)} \tag{1}$$

Here,

j - varies from 0 to 2

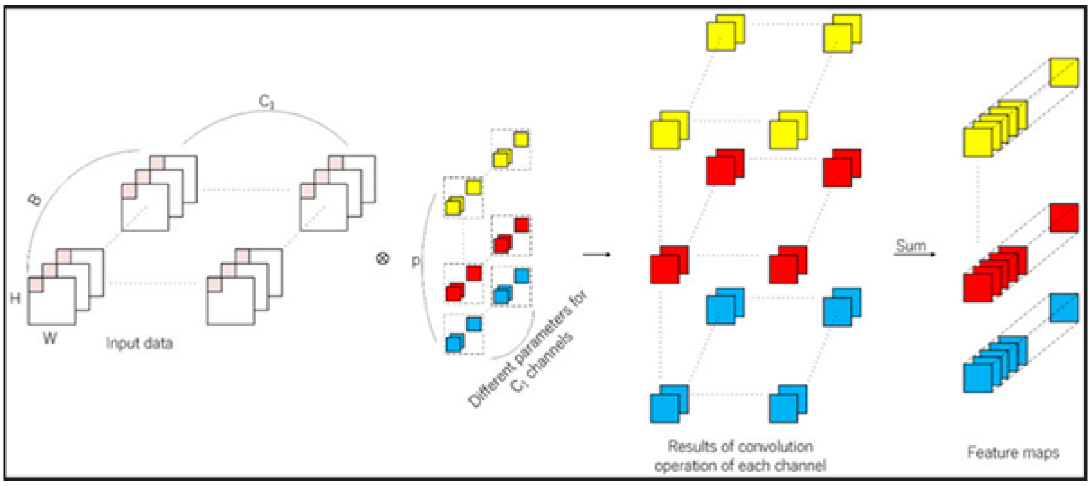
k - varies from 48 to 68

(x,y,z) is the position of the kth feature in the jth convolutional layer.

h - height

w - weight

In the 3D-CNN-based classification task, let the input data dimension, convolution kernel size, and the kernel number be W × H × B × C1, k × k × k, p, respectively. If padding is not used and the stride is 1, then the dimension of the feature map is (W-k + 1)(H-k + 1)(B-k + 1)p generated by the 3D convolutional layer. In terms of network parameters, the number of weight parameters is p × k × k × k × C1 for the 3D convolutional layer. Therefore, on the one hand, the feature map generated by the 3D convolution operation contains more spectral information. Figure 3.5 shows the step by step operation of CNN model.

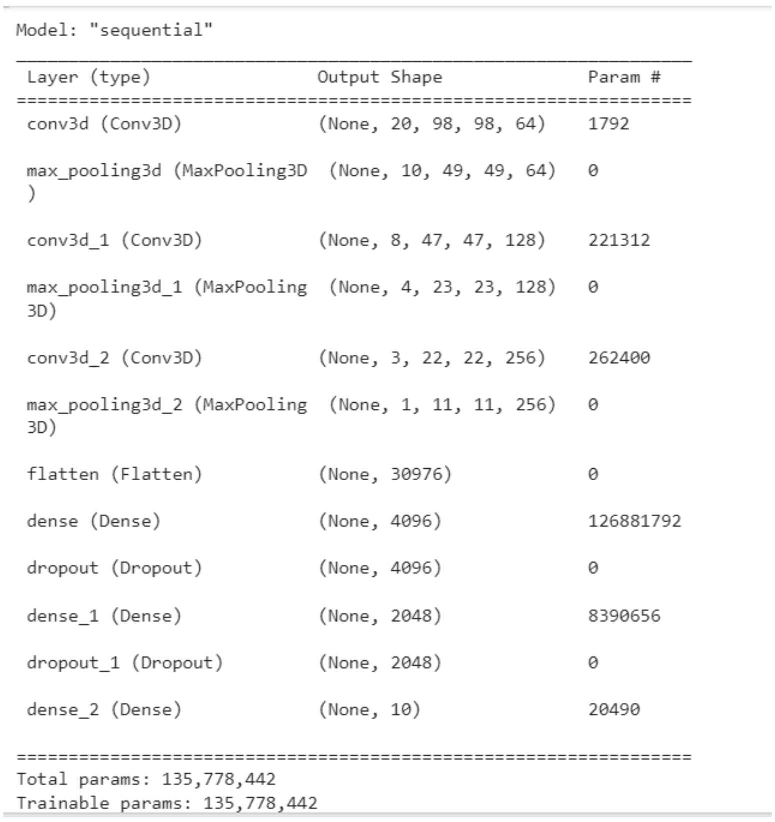


C_1
B
H
W
Input data
⊗
p
Different parameters for C_1 channels
Results of convolution operation of each channel
Sum
Feature maps

Figure 3.5: **Step by step CNN model operation**

The overall 3D convolution operation. The input data dimension is W × H × B × C1, where B is the band number and C1 is the channel number; the 3D convolution kernel size is k × k × k and the last k denotes the coverage of convolution kernel over spectral dimension in each convolution operation; if padding is not used and the stride is 1, then the output data dimension of single kernel is (W - k + 1)(H - k + 1)(B k + 1).

The model structure used in our project used for English dataset is shown in Figure 3.6.

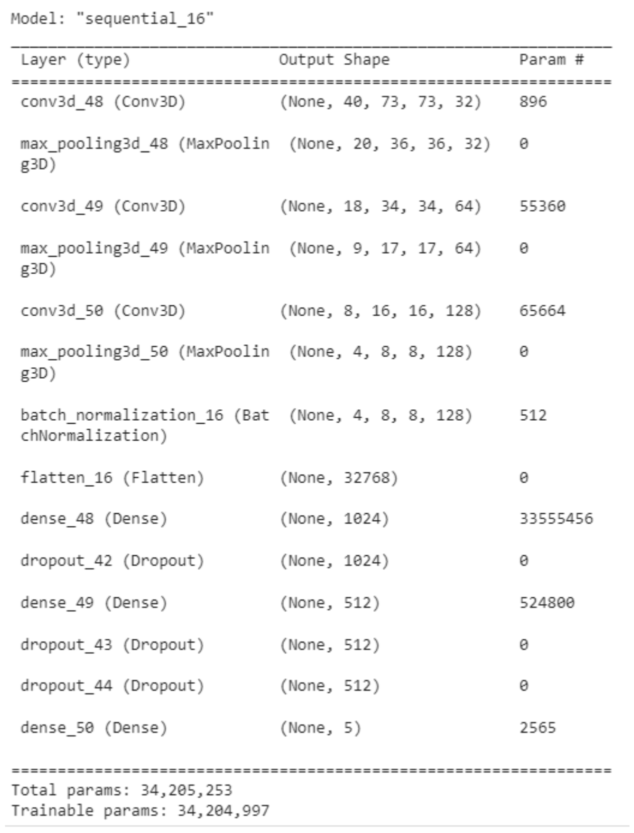


```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv3d (Conv3D)             (None, 20, 98, 98, 64)    1792

 max_pooling3d (MaxPooling3D  (None, 10, 49, 49, 64)   0
 )

 conv3d_1 (Conv3D)           (None, 8, 47, 47, 128)    221312

 max_pooling3d_1 (MaxPooling  (None, 4, 23, 23, 128)   0
 3D)

 conv3d_2 (Conv3D)           (None, 3, 22, 22, 256)    262400

 max_pooling3d_2 (MaxPooling  (None, 1, 11, 11, 256)   0
 3D)

 flatten (Flatten)           (None, 30976)             0

 dense (Dense)               (None, 4096)              126881792

 dropout (Dropout)           (None, 4096)              0

 dense_1 (Dense)             (None, 2048)              8390656

 dropout_1 (Dropout)         (None, 2048)              0

 dense_2 (Dense)             (None, 10)                20490

=================================================================
Total params: 135,778,442
Trainable params: 135,778,442
```

Figure 3.6: **Model structure of CNN 3D model for English Dataset**

The model structure used in our project for Kannada dataset is shown in Figure 3.7
Figure 3.8 shows the summary of model structure of CNN 3D for English Dataset.
Figure 3.9 shows the summary of model structure of CNN 3D for Kannada Dataset.

In CNN 3D model we have used three convolution layers and 3 fully connected layers along with we have also used different activation functions for different layers.We have kept pool_size and kernal_size = 3. The three filters which we will be using are set to 64,128 and 256 respectively. We will be using dropout. Dropout is very essential as it helps to avoid overfitting. We have set the dropout value to 0.5. For hidden layers, we will be using the activation function named “RELU”. Similarly, for the output layer, we will be using the activation function named “SOFTMAX”.
Dense Layer is simple layer of neurons in which each neuron receives input from all the neurons of previous layer, thus called as dense. Dense Layer is used to classify image based on output from convolutional layers.



```
Model: "sequential_16"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv3d_48 (Conv3D)          (None, 40, 73, 73, 32)    896

 max_pooling3d_48 (MaxPoolin  (None, 20, 36, 36, 32)   0
 g3D)

 conv3d_49 (Conv3D)          (None, 18, 34, 34, 64)    55360

 max_pooling3d_49 (MaxPoolin  (None, 9, 17, 17, 64)    0
 g3D)

 conv3d_50 (Conv3D)          (None, 8, 16, 16, 128)    65664

 max_pooling3d_50 (MaxPoolin  (None, 4, 8, 8, 128)     0
 g3D)

 batch_normalization_16 (Bat  (None, 4, 8, 8, 128)     512
 chNormalization)

 flatten_16 (Flatten)        (None, 32768)             0

 dense_48 (Dense)            (None, 1024)              33555456

 dropout_42 (Dropout)        (None, 1024)              0

 dense_49 (Dense)            (None, 512)               524800

 dropout_43 (Dropout)        (None, 512)               0

 dropout_44 (Dropout)        (None, 512)               0

 dense_50 (Dense)            (None, 5)                 2565

=================================================================
Total params: 34,205,253
Trainable params: 34,204,997
```

Figure 3.7: **Model structure of CNN 3D model for Kannada Dataset**

There are several layers in CNN model. They are:

1. **Convolutional:**
   Convolutional layers consist of a rectangular grid of neurons. It requires that the previous layer also be a rectangular grid of neurons. Each neuron takes inputs from a rectangular section of the previous layer; the weights for this rectangular section are the same for each neuron in the convolutional layer. Thus, the convolutional layer is just an image convolution of the previous layer, where the weights specify the convolution filter.

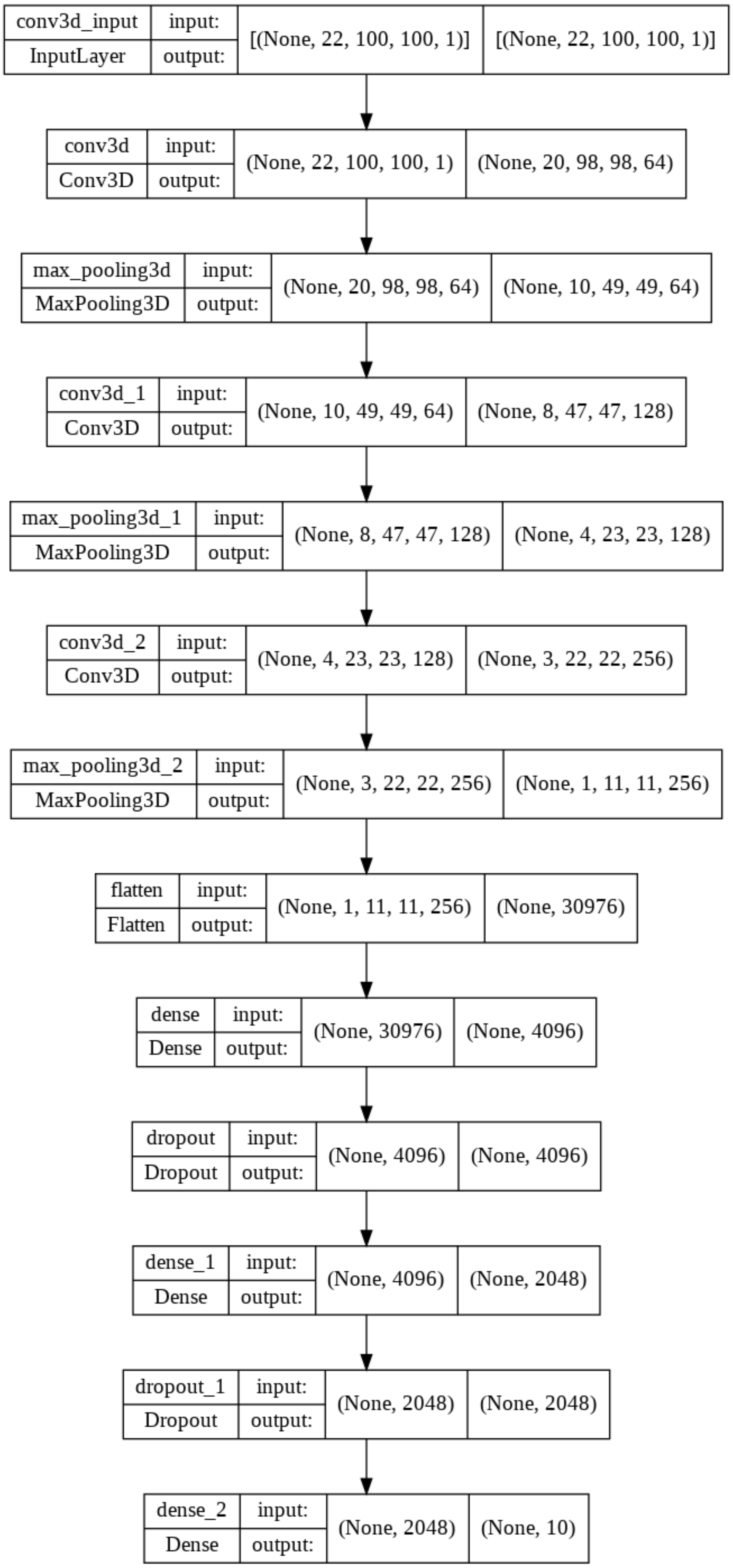
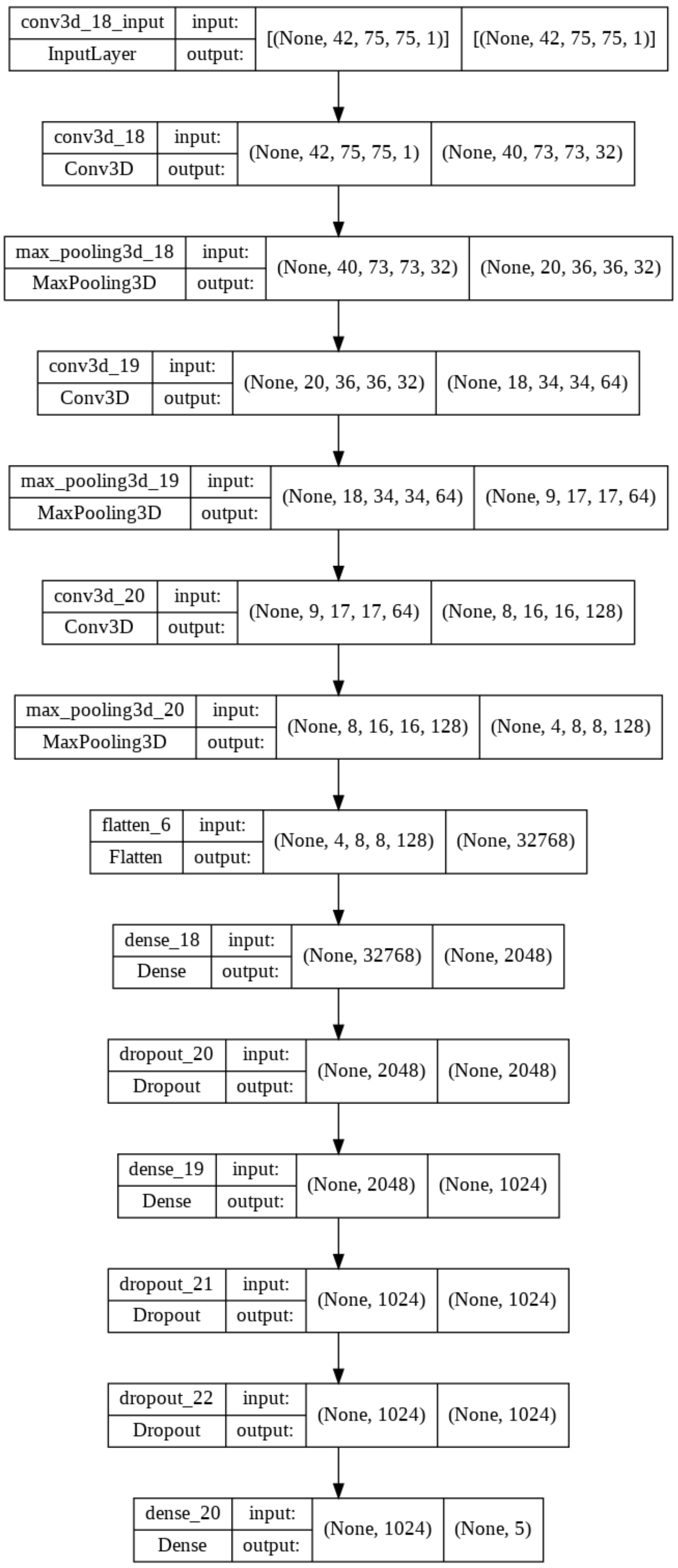

conv3d_input | InputLayer | input: [(None, 22, 100, 100, 1)] | output: [(None, 22, 100, 100, 1)]
conv3d | Conv3D | input: (None, 22, 100, 100, 1) | output: (None, 20, 98, 98, 64)
max_pooling3d | MaxPooling3D | input: (None, 20, 98, 98, 64) | output: (None, 10, 49, 49, 64)
conv3d_1 | Conv3D | input: (None, 10, 49, 49, 64) | output: (None, 8, 47, 47, 128)
max_pooling3d_1 | MaxPooling3D | input: (None, 8, 47, 47, 128) | output: (None, 4, 23, 23, 128)
conv3d_2 | Conv3D | input: (None, 4, 23, 23, 128) | output: (None, 3, 22, 22, 256)
max_pooling3d_2 | MaxPooling3D | input: (None, 3, 22, 22, 256) | output: (None, 1, 11, 11, 256)
flatten | Flatten | input: (None, 1, 11, 11, 256) | output: (None, 30976)
dense | Dense | input: (None, 30976) | output: (None, 4096)
dropout | Dropout | input: (None, 4096) | output: (None, 4096)
dense_1 | Dense | input: (None, 4096) | output: (None, 2048)
dropout_1 | Dropout | input: (None, 2048) | output: (None, 2048)
dense_2 | Dense | input: (None, 2048) | output: (None, 10)

Figure 3.8: **Summary of model structure of CNN 3D for English Dataset**



conv3d_18_input | InputLayer | input: | output: | [(None, 42, 75, 75, 1)] | [(None, 42, 75, 75, 1)]
conv3d_18 | Conv3D | input: | output: | (None, 42, 75, 75, 1) | (None, 40, 73, 73, 32)
max_pooling3d_18 | MaxPooling3D | input: | output: | (None, 40, 73, 73, 32) | (None, 20, 36, 36, 32)
conv3d_19 | Conv3D | input: | output: | (None, 20, 36, 36, 32) | (None, 18, 34, 34, 64)
max_pooling3d_19 | MaxPooling3D | input: | output: | (None, 18, 34, 34, 64) | (None, 9, 17, 17, 64)
conv3d_20 | Conv3D | input: | output: | (None, 9, 17, 17, 64) | (None, 8, 16, 16, 128)
max_pooling3d_20 | MaxPooling3D | input: | output: | (None, 8, 16, 16, 128) | (None, 4, 8, 8, 128)
flatten_6 | Flatten | input: | output: | (None, 4, 8, 8, 128) | (None, 32768)
dense_18 | Dense | input: | output: | (None, 32768) | (None, 2048)
dropout_20 | Dropout | input: | output: | (None, 2048) | (None, 2048)
dense_19 | Dense | input: | output: | (None, 2048) | (None, 1024)
dropout_21 | Dropout | input: | output: | (None, 1024) | (None, 1024)
dropout_22 | Dropout | input: | output: | (None, 1024) | (None, 1024)
dense_20 | Dense | input: | output: | (None, 1024) | (None, 5)

Figure 3.9: **Summary of model structure of CNN 3D for Kannada Dataset**

In addition, there may be several grids in each convolutional layer; each grid takes inputs from all the grids in the previous layer, using potentially different filters. Suppose that we have some N×N square neuron layer which is followed by our convolutional layer. If we use an m×m filter , our convolutional layer output will be of size (N-m+1)×(N-m+1). In order to compute the pre-nonlinearity input to some unit in our layer, we need to sum up the contributions (weighted by the filter components) from the previous layer cells:

$$x^l_{(i,j)} = \sum_{(a=0)}^{(m-1)} \sum_{(a=0)}^{(m-1)} w_{ab} y^{(l-1)}_{((i+a)(j+b))} \tag{2}$$

Then, the convolutional layer applies its nonlinearity:

$$y^l_{ij} = \sigma(x^l_{ij}) \tag{3}$$

2. **Max- Pooling:**

After each convolutional layer, there may be a pooling layer. The pooling layer takes small rectangular blocks from the convolutional layer and subsamples it to produce a single output from that block. There are several ways to do this pooling, such as taking the average or the maximum, or a learned linear combination of the neurons in the block. Our pooling layers will always be max-pooling layers; that is, they take the maximum of the block they are pooling.
The max-pooling layers are quite simple, and do not teach themselves. They simply take some k×k region and output a single value, which is the maximum in that region. For 25 instance, if their input layer is a N×N layer, they will then output a Nk×Nk layer, as each k×k block is reduced to just a single value via the max function.
Assume we have an input volume of width W^1, height H^1, and depth D^1. The pooling layer requires 2 hyper parameters, kernel/filter size F and stride S.
On applying the pooling layer over the input volume, output dimensions of output volume will be

$$W^2 = \frac{(W^1 - F)}{S} + 1 \tag{4}$$

$$H^2 = \frac{(H^1 - F)}{S} + 1 \tag{5}$$

$$D^2 = D^1 \tag{6}$$

3. **Fully-Connected:**

   Finally, after several convolutional and max pooling layers, the high-level reasoning in the neural network is done via fully connected layers. A fully connected layer takes all neurons in the previous layer (be it fully connected, pooling, or convolutional) and connects it to every single neuron it has. Fully connected layers are not spatially located anymore (you can visualize them as one-dimensional), so there can be no convolutional layers after a fully connected layer.

   A fully connected layer is a dot product between a data matrix and a weights matrix.

$$y = X.W \tag{7}$$

   The data will have an input shape of (batch_size, n_features) and the weight matrix will have shape (n_features, n_units). This equation is important, because it shows you that the output shape will be (batch_size, n_units).When we want to add an n_units bias vector Then the equation will be

$$y = XW + b \tag{8}$$

   If we let the activation be an arbitrary function, f, then the full operation can be written:

$$y = f(XW + b) \tag{9}$$

   During training, W and b will both be learned in order to best fit the data, (X, y)

   Activation Function: Activation functions we used in our model are ReLU activation functions. ReLU stands for rectified linear activation unit and is considered one of the few milestones in the deep learning revolution. It is simple yet really better than its predecessor activation functions such as sigmoid or tanh. The formula for the ReLU activation function is

$$f(x) = max(0, x) \tag{10}$$

   The ReLU function is its derivative and both are monotonic. The function returns 0 if it receives any negative input, but for any positive value x, it returns that value back. Thus it gives an output that has a range from 0 to infinity.The Figure 3.10 shows the activation curve.

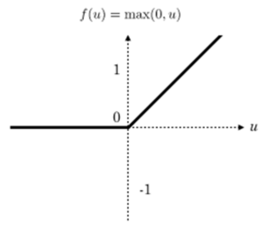


Figure 3.10: **ReLU activation curve**

3.3.5 Visual Speech Recognition Using LSTM Model

Long Short-Term Memory (LSTM) networks are a type of recurrent neural network capable of learning order dependence in sequence prediction problems.This is a behavior required in complex problem domains like machine translation, speech recognition, and more.LSTM has a chain structure that contains four neural networks and different memory blocks called cells.Information is retained by the cells and the memory manipulations are done by the gates. There are three gates – Forget Gate, Input Gate and Output Gate.

The model which is created contains an LSTM layer with 128 hidden units and 8 timestamps as the first layer.The operations inside LSTM cell are as follows

$$c\tilde{}^{<t>} = tanh(W_c[a^{<t-1>}, x^{<t>}] + b_c) \tag{11}$$

$$\Gamma_u = (W_u[a^{<t-1>}, x^{<t>}] + b_u) \tag{12}$$

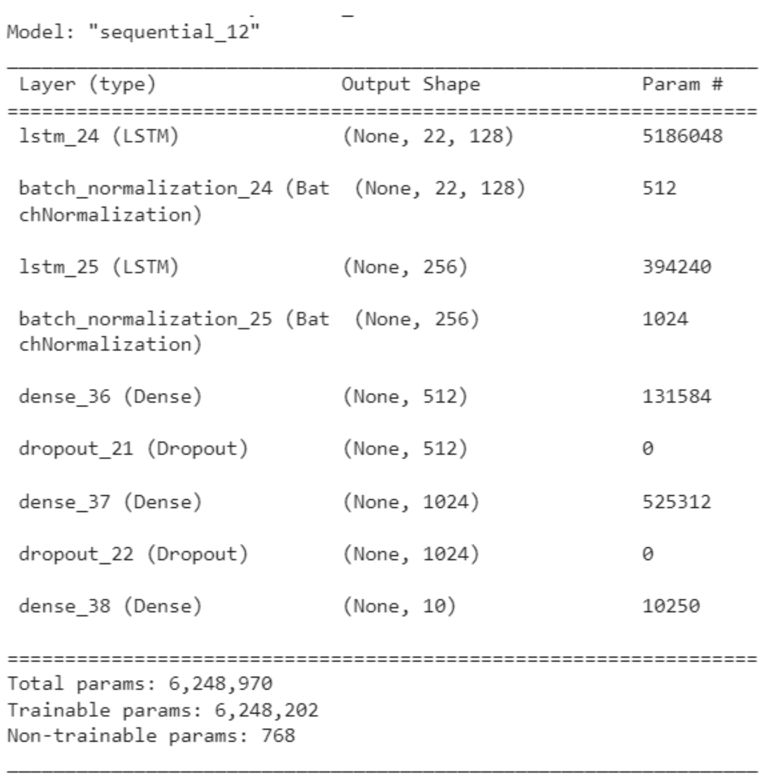
$$\Gamma_f = (W_f[a^{<t-1>}, x^{<t>}] + b_f) \tag{13}$$

$$\Gamma_o = (W_o[a^{<t-1>}, x^{<t>}] + b_o) \tag{14}$$

$$c^{<t>} = \Gamma_u c\tilde{}^{<t-1>} + \Gamma_f * c^{<t-1>} \tag{15}$$

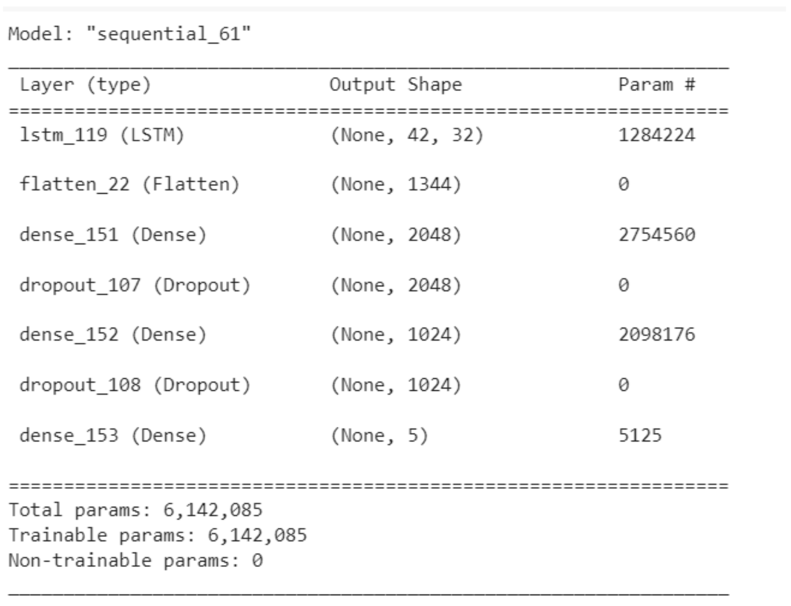
$$a^{<t>} = \Gamma_o * tanh(c^{<t>}) \tag{16}$$

Where c represents the memory cell, t represents timestamp, represents the update gate. frepresents the forget gate, represents the output gate, represents the sigmoid function, Wu represents the weights of update gate, Wf represents the weights of forget gate, which represents the weights of output gate, b represents bias, represents candidate cell variables. The model structure of LSTM is shown in the figure 3.11 and the model summary of LSTM model for English Dataset is shown in figure 3.13.The model structure of LSTM is shown



```
Model: "sequential_12"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_24 (LSTM)              (None, 22, 128)           5186048

 batch_normalization_24 (Bat  (None, 22, 128)          512
 chNormalization)

 lstm_25 (LSTM)              (None, 256)               394240

 batch_normalization_25 (Bat  (None, 256)              1024
 chNormalization)

 dense_36 (Dense)            (None, 512)               131584

 dropout_21 (Dropout)        (None, 512)               0

 dense_37 (Dense)            (None, 1024)              525312

 dropout_22 (Dropout)        (None, 1024)              0

 dense_38 (Dense)            (None, 10)                10250

=================================================================
Total params: 6,248,970
Trainable params: 6,248,202
Non-trainable params: 768
_________________________________________________________________
```

Figure 3.11: **Model structure for LSTM model for English Dataset**



```
Model: "sequential_61"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_119 (LSTM)             (None, 42, 32)            1284224

 flatten_22 (Flatten)        (None, 1344)              0

 dense_151 (Dense)           (None, 2048)              2754560

 dropout_107 (Dropout)       (None, 2048)              0

 dense_152 (Dense)           (None, 1024)              2098176

 dropout_108 (Dropout)       (None, 1024)              0

 dense_153 (Dense)           (None, 5)                 5125

=================================================================
Total params: 6,142,085
Trainable params: 6,142,085
Non-trainable params: 0
_________________________________________________________________
```

Figure 3.12: **Model structure for LSTM model for Kannada Dataset**

in the Figure 3.12 and the model summary of LSTM model for Kannada Dataset is shown in Figure 3.14.

A typical LSTM network is comprised of different memory blocks called cells (the rectangles that we see in the Figure 3.15). There are two states that are being transferred to the

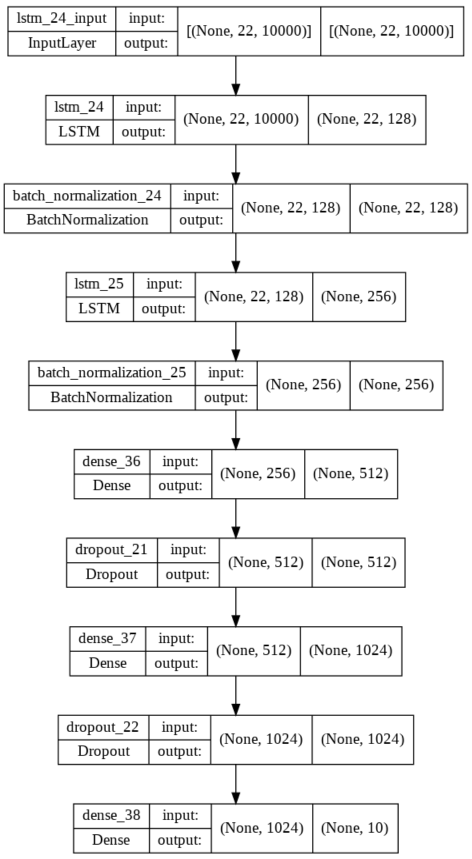

lstm_24_input | InputLayer | input: [(None, 22, 10000)] | output: [(None, 22, 10000)]
lstm_24 | LSTM | input: (None, 22, 10000) | output: (None, 22, 128)
batch_normalization_24 | BatchNormalization | input: (None, 22, 128) | output: (None, 22, 128)
lstm_25 | LSTM | input: (None, 22, 128) | output: (None, 256)
batch_normalization_25 | BatchNormalization | input: (None, 256) | output: (None, 256)
dense_36 | Dense | input: (None, 256) | output: (None, 512)
dropout_21 | Dropout | input: (None, 512) | output: (None, 512)
dense_37 | Dense | input: (None, 512) | output: (None, 1024)
dropout_22 | Dropout | input: (None, 1024) | output: (None, 1024)
dense_38 | Dense | input: (None, 1024) | output: (None, 10)

Figure 3.13: **Summary of model structure of LSTM model for English Dataset.**

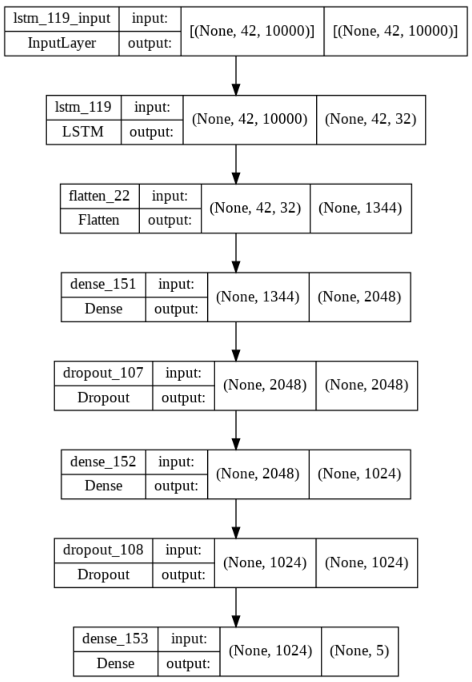

lstm_119_input | InputLayer | input: [(None, 42, 10000)] | output: [(None, 42, 10000)]
lstm_119 | LSTM | input: (None, 42, 10000) | output: (None, 42, 32)
flatten_22 | Flatten | input: (None, 42, 32) | output: (None, 1344)
dense_151 | Dense | input: (None, 1344) | output: (None, 2048)
dropout_107 | Dropout | input: (None, 2048) | output: (None, 2048)
dense_152 | Dense | input: (None, 2048) | output: (None, 1024)
dropout_108 | Dropout | input: (None, 1024) | output: (None, 1024)
dense_153 | Dense | input: (None, 1024) | output: (None, 5)

Figure 3.14: **Summary of model structure of LSTM model for Kannada Dataset.**

next cell; the cell state and the hidden state. The memory blocks are responsible for remembering things and manipulations to this memory is done through three major mechanisms, called gates.

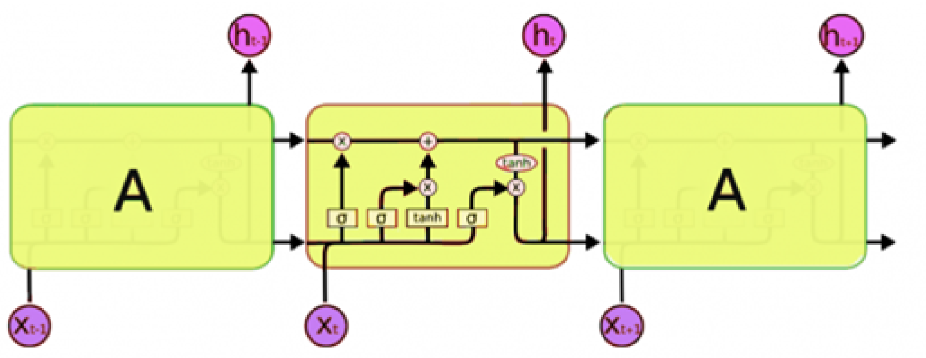


Figure 3.15: **Internal structure of LSTM cell**

1. **Forget Gate:**

   A forget gate is responsible for removing information from the cell state. The information that is no longer required for the LSTM to understand things or the information that is of less importance is removed via multiplication of a filter. This is required for optimizing the performance of the LSTM network.The figure 3.16 shows the structure of Forget gate in LSTM model.

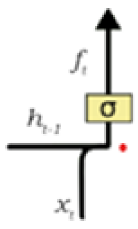


Figure 3.16: **Structure of Forget Gate**

   This gate takes in two inputs; ht-1 and xt. ht-1 is the hidden state from the previous cell or the output of the previous cell and xt is the input at that particular time step. The given inputs are multiplied by the weight matrices and a bias is added. Following this, the sigmoid function is applied to this value. The sigmoid function outputs a vector, with values ranging from 0 to 1, corresponding to each number in the cell state. Basically, the sigmoid function is responsible for deciding which values to keep and which to discard. If a ‘0’ is output for a particular value in the cell state, it means that the forget gate wants the cell state to forget that piece of information completely. Similarly, a '1' means that the forget gate wants to remember that entire piece of information. This vector output from the sigmoid function is multiplied to the cell state.

2. **Input Gate:**
   The input gate is responsible for the addition of information to the cell state. Figure 3.17 is the structure of Input Gate

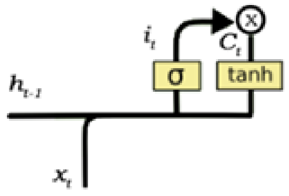


Figure 3.17: **Structure of Input Gate**

   (a) Regulating what values need to be added to the cell state by involving a sigmoid function. This is basically very similar to the forget gate and acts as a filter for all the information from ht-1 and xt.
   (b) Creating a vector containing all possible values that can be added (as perceived from ht-1 and xt) to the cell state. This is done using the tanh function, which outputs values from -1 to +1.
   (c) Multiplying the value of the regulatory filter (the sigmoid gate) to the created vector (the tanh function) and then adding this useful information to the cell state via addition operation.

3. **Output Gate:**
   This job of selecting useful information from the current cell state and showing it out as an output is done via the output gate.The output gate is shown in the figure 3.18.

   The functioning of an output gate can again be broken down to three steps:

   (a) Creating a vector after applying tanh function to the cell state, thereby scaling the values to the range -1 to +1.

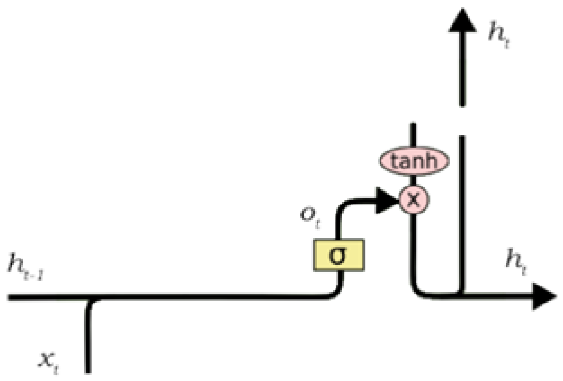


Figure 3.18: **Structure of Output Gate**

(b) Making a filter using the values of ht-1 and xt, such that it can regulate the values that need to be output from the vector created above. This filter again employs a sigmoid function.

(c) Multiplying the value of this regulatory filter to the vector created in step 1, and sending it out as output and also to the hidden state of the next cell.

Batch Normalization:

Batch normalization is a method we can use to normalize the inputs of each layer, in order to fight the internal covariate shift problem.During training time, a batch normalization layer does the following: Calculate the mean and variance of the layers input.

$$\mu_B = \frac{1}{m}(\sum_{(i=1)}^{m} x_i) \tag{17}$$

$$\sigma_B^2 = \frac{1}{m}(\sum_{(i=1)}^{m} (x_i - \mu_B^2)) \tag{18}$$

Normalize the layer inputs using the previously calculated batch statistics.

$$x_i' = x_i - \frac{\mu_B}{\sqrt{(\sigma_B^2 + \varepsilon)}} \tag{19}$$

Scale and shift in order to obtain the output of the layer.

$$y_i = \gamma(x_i') + \beta \tag{20}$$

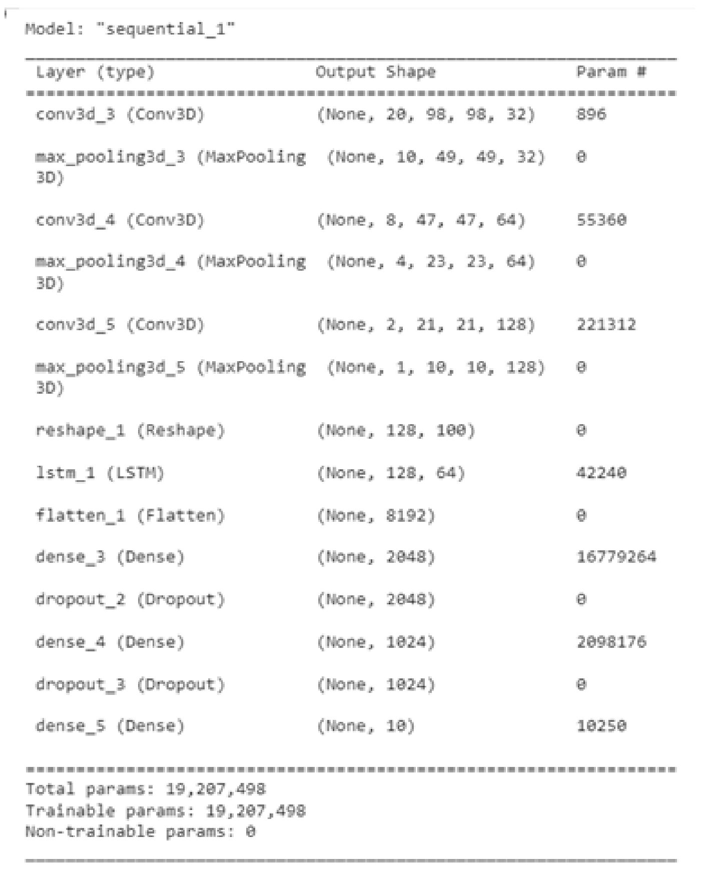
Dense Layer:

The dense layer's neuron in a model receives output from every neuron of its preceding layer, where neurons of the dense layer perform matrix-vector multiplication. Matrix vector multiplication is a procedure where the row vector of the output from the preceding layers is equal to the column vector of the dense layer. The general rule of matrix-vector multiplication is that the row vector must have as many columns like the column vector.

Drop out layer is used to reduce the over fitting. If over fitting is increased there will be a non-linearity between training and validation dataset. To reduce this non linearity, drop out layers are used.

3.3.6 Visual Speech Recognition Using Hybrid Model

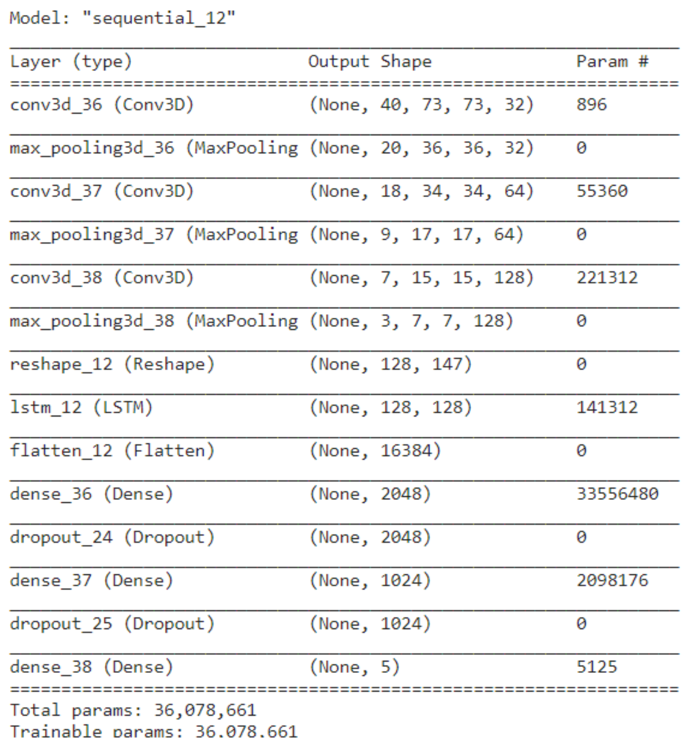
Figure 3.19 shows the structure of hybrid model implemented using CNN and LSTM.



```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv3d_3 (Conv3D)           (None, 20, 98, 98, 32)    896

 max_pooling3d_3 (MaxPooling  (None, 10, 49, 49, 32)   0
 3D)

 conv3d_4 (Conv3D)           (None, 8, 47, 47, 64)     55360

 max_pooling3d_4 (MaxPooling  (None, 4, 23, 23, 64)    0
 3D)

 conv3d_5 (Conv3D)           (None, 2, 21, 21, 128)    221312

 max_pooling3d_5 (MaxPooling  (None, 1, 10, 10, 128)   0
 3D)

 reshape_1 (Reshape)         (None, 128, 100)          0

 lstm_1 (LSTM)               (None, 128, 64)           42240

 flatten_1 (Flatten)         (None, 8192)              0

 dense_3 (Dense)             (None, 2048)              16779264

 dropout_2 (Dropout)         (None, 2048)              0

 dense_4 (Dense)             (None, 1024)              2098176

 dropout_3 (Dropout)         (None, 1024)              0

 dense_5 (Dense)             (None, 10)                10250

=================================================================
Total params: 19,207,498
Trainable params: 19,207,498
Non-trainable params: 0
_________________________________________________________________
```

Figure 3.19: **Model structure of (CNN 3D + LSTM) hybrid model for English Dataset**

Figure 3.20 shows the structure of hybrid model implemented using CNN and LSTM.



```
Model: "sequential_12"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv3d_36 (Conv3D)           (None, 40, 73, 73, 32)    896
_________________________________________________________________
max_pooling3d_36 (MaxPooling (None, 20, 36, 36, 32)    0
_________________________________________________________________
conv3d_37 (Conv3D)           (None, 18, 34, 34, 64)    55360
_________________________________________________________________
max_pooling3d_37 (MaxPooling (None, 9, 17, 17, 64)     0
_________________________________________________________________
conv3d_38 (Conv3D)           (None, 7, 15, 15, 128)    221312
_________________________________________________________________
max_pooling3d_38 (MaxPooling (None, 3, 7, 7, 128)      0
_________________________________________________________________
reshape_12 (Reshape)         (None, 128, 147)          0
_________________________________________________________________
lstm_12 (LSTM)               (None, 128, 128)          141312
_________________________________________________________________
flatten_12 (Flatten)         (None, 16384)             0
_________________________________________________________________
dense_36 (Dense)             (None, 2048)              33556480
_________________________________________________________________
dropout_24 (Dropout)         (None, 2048)              0
_________________________________________________________________
dense_37 (Dense)             (None, 1024)              2098176
_________________________________________________________________
dropout_25 (Dropout)         (None, 1024)              0
_________________________________________________________________
dense_38 (Dense)             (None, 5)                 5125
=================================================================
Total params: 36,078,661
Trainable params: 36,078,661
```

Figure 3.20: **Model structure of (CNN 3D + LSTM) hybrid model for Kannada Dataset**

CNN–LSTM was developed for visual time series prediction problems and generating textual descriptions from the sequences of images. The CNN–LSTM architecture uses CNN layers for feature extraction on input data and combines with LSTM to support sequence prediction. Specifically, CNN extracts the features from spatial inputs and uses them in the LSTM architecture to output the caption.The CNN-LSTM model architecture is shown in figure 3.21.

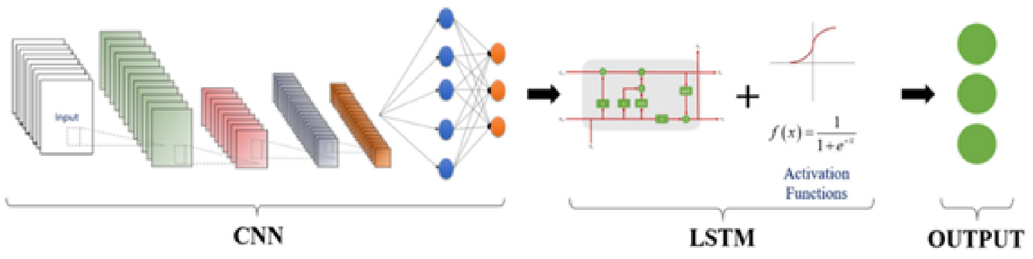


Figure 3.21: **Architecture of CNN-LSTM model**

The Figure 3.22 shows the summary of model structure of CNN-LSTM Hybrid model for English Dataset.

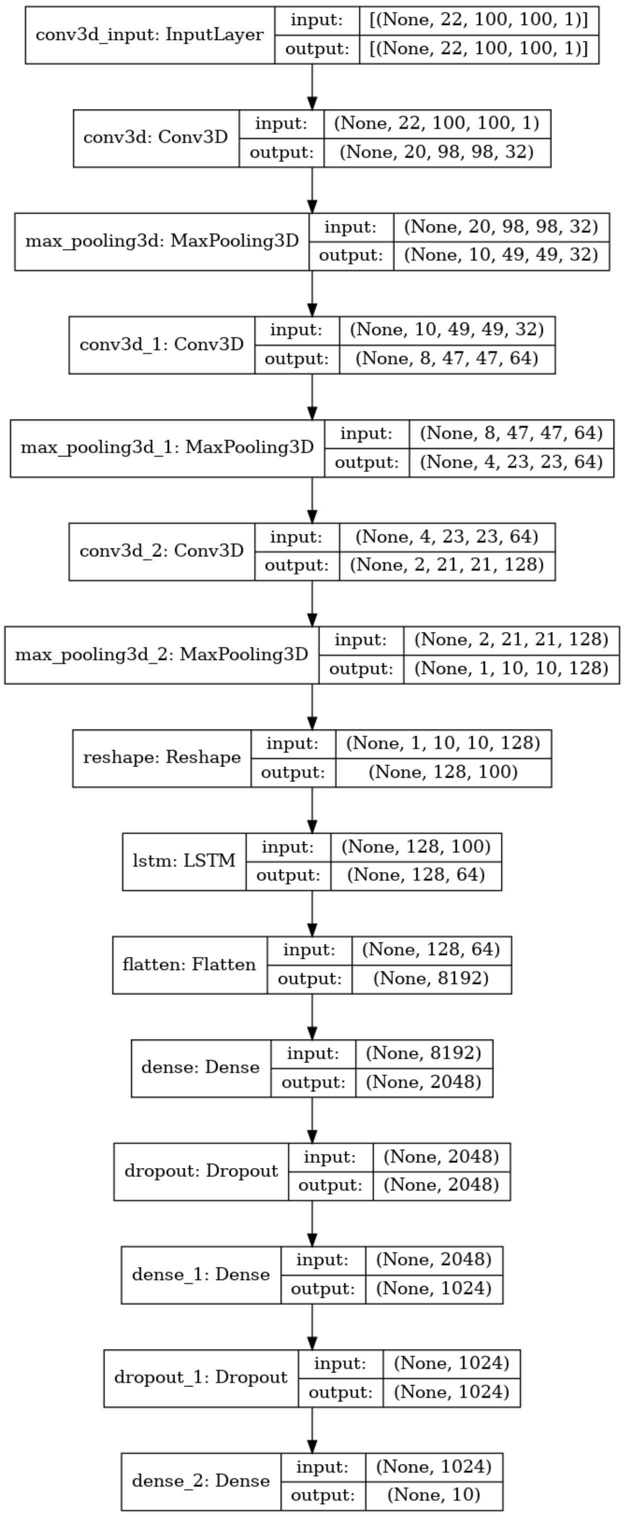

conv3d_input: InputLayer
input: [(None, 22, 100, 100, 1)]
output: [(None, 22, 100, 100, 1)]
conv3d: Conv3D
input: (None, 22, 100, 100, 1)
output: (None, 20, 98, 98, 32)
max_pooling3d: MaxPooling3D
input: (None, 20, 98, 98, 32)
output: (None, 10, 49, 49, 32)
conv3d_1: Conv3D
input: (None, 10, 49, 49, 32)
output: (None, 8, 47, 47, 64)
max_pooling3d_1: MaxPooling3D
input: (None, 8, 47, 47, 64)
output: (None, 4, 23, 23, 64)
conv3d_2: Conv3D
input: (None, 4, 23, 23, 64)
output: (None, 2, 21, 21, 128)
max_pooling3d_2: MaxPooling3D
input: (None, 2, 21, 21, 128)
output: (None, 1, 10, 10, 128)
reshape: Reshape
input: (None, 1, 10, 10, 128)
output: (None, 128, 100)
lstm: LSTM
input: (None, 128, 100)
output: (None, 128, 64)
flatten: Flatten
input: (None, 128, 64)
output: (None, 8192)
dense: Dense
input: (None, 8192)
output: (None, 2048)
dropout: Dropout
input: (None, 2048)
output: (None, 2048)
dense_1: Dense
input: (None, 2048)
output: (None, 1024)
dropout_1: Dropout
input: (None, 1024)
output: (None, 1024)
dense_2: Dense
input: (None, 1024)
output: (None, 10)

Figure 3.22: **Summary of Model structure of hybrid model (CNN 3D + LSTM) for English Dataset**

The Figure 3.23 shows the summary of model structure of CNN-LSTM Hybrid model for Kannada Dataset.

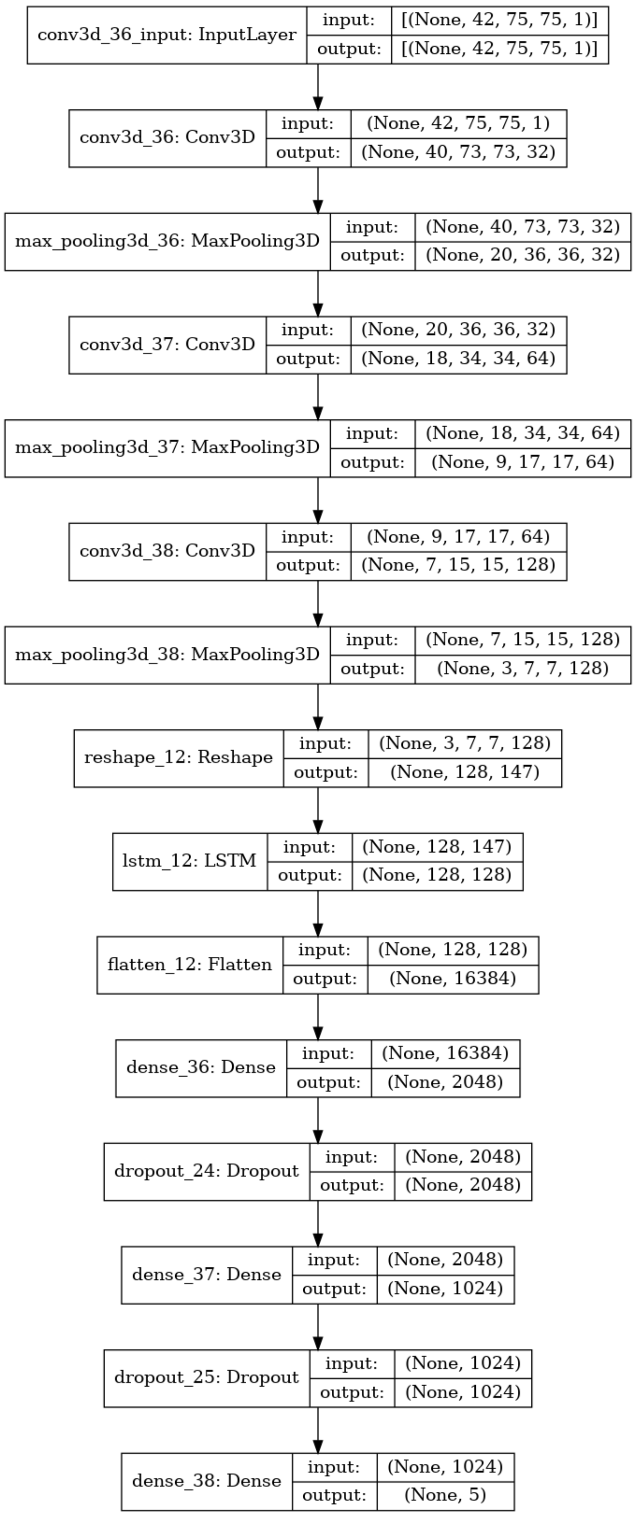

conv3d_36_input: InputLayer
input: [(None, 42, 75, 75, 1)]
output: [(None, 42, 75, 75, 1)]
conv3d_36: Conv3D
input: (None, 42, 75, 75, 1)
output: (None, 40, 73, 73, 32)
max_pooling3d_36: MaxPooling3D
input: (None, 40, 73, 73, 32)
output: (None, 20, 36, 36, 32)
conv3d_37: Conv3D
input: (None, 20, 36, 36, 32)
output: (None, 18, 34, 34, 64)
max_pooling3d_37: MaxPooling3D
input: (None, 18, 34, 34, 64)
output: (None, 9, 17, 17, 64)
conv3d_38: Conv3D
input: (None, 9, 17, 17, 64)
output: (None, 7, 15, 15, 128)
max_pooling3d_38: MaxPooling3D
input: (None, 7, 15, 15, 128)
output: (None, 3, 7, 7, 128)
reshape_12: Reshape
input: (None, 3, 7, 7, 128)
output: (None, 128, 147)
lstm_12: LSTM
input: (None, 128, 147)
output: (None, 128, 128)
flatten_12: Flatten
input: (None, 128, 128)
output: (None, 16384)
dense_36: Dense
input: (None, 16384)
output: (None, 2048)
dropout_24: Dropout
input: (None, 2048)
output: (None, 2048)
dense_37: Dense
input: (None, 2048)
output: (None, 1024)
dropout_25: Dropout
input: (None, 1024)
output: (None, 1024)
dense_38: Dense
input: (None, 1024)
output: (None, 5)

Figure 3.23: **Summary of Model structure of hybrid model (CNN 3D + LSTM) for Kannada Dataset**

This mathematical formulation is commonly used to indicate the equation of a convolutional layer:

$$X_j^l = f(\sum_{(ie,M_j)} (X_i^{(l-1)} * k_{ij}^l) + b_i^l) \tag{22}$$

In order to lower the complicated resolution of each feature map, this layer employs a mix of sub sampling and local averaging. It also disregards output reactivity. The mathematical formulation of this layer is given below.

$$X_j^l = f(B_{jdown}^l(X_j^{(l-1)}) + b_i^l) \tag{23}$$

The mathematical form of LSTM is shown in following equations:

$$f_t = ReLu(w_f.(h_{(t-1)}, x_t) + b_f) \tag{24}$$

$$i_t = ReLu(w_i.(h_{(t-1)}, x_t) + b_t) \tag{25}$$

$$c_t' = ReLu(w_c.(h_{(t-1)}, x_t) + b_c) \tag{26}$$

$$O_t = ReLu(w_o.(h_{(t-1)}, x_t) + b_t) \tag{27}$$

$$c_t = f_t.c_{(t-1)} + i_t.c_t' \tag{28}$$

$$h_t = L.ReLu(c_t) \tag{29}$$

In CNN 3D model we have used three convolution layers and 3 fully connected layers along with We have also used different activation functions for different layers.We have kept pool_size and kernal_size = 3. The three filters which we will be using are set to 64,128 and 256 respectively and we have used a single LSTM model with filter size of 128. We will be using dropout. Dropout is very essential as it helps to avoid over fitting. We have set the dropout value to 0.5. For hidden layers, we will be using the activation function named “RELU”. Similarly, for the output layer, we will be using the activation function named “SOFTMAX”.

Input to the 1st layer 3D CNN is 3D tensors. The CNN feature extraction model made by three 3D convolutional layers. We incorporated the MaxPooling layer and Rectified Linear Unit (ReLU) layer in between the two consecutive convolution layers. The CNN 3D model usually gives feature map as output,but it has limitation that it knows the exact location of the features in the input. It means that little disturbance in the place of the feature in the input will lead to a different feature map.Then a pooling layer is added after the convolutional layer for making the model less dependent on feature map disturbance .

Then activation function in our case ‘ReLU’ is used to increase the capabitlity of model to learn in complex structure too.After input weights enters into LSTM model the dimension of tensors get changed from 3d to 2d and gets trained intoLSTM model. LSTM use a series of gates which control how the information in a sequence of data comes into,is stored in and leaves the network. The LSTM gets the target class in the sense name of the word in our case as output.

Drop out layer is used to reduce the over fitting. If over fitting is increased there will be a non-linearity between training and validation dataset. To reduce this non linearity, drop out layers are used. Dense Layer is simple layer of neurons in which each neuron receives input from all the neurons of previous layer, thus called as dense. Dense Layer is used to classify image based on output from convolutional layers.

Chapter 4

Results and Discussion

This Chapter includes intermediate and final results obtained while developing the model for Visual Speech Recognition using Hybrid model. First the results for the CNN model are obtained and shown in the below sections. Then the results of Visual Speech Recognition for LSTM model are shown. Then the results for the hybrid model which is obtained by combining the CNN and LSTM models is shown in this chapter.

4.1 Result for Visual Speech Recognition Using CNN 3D Model

Dataset Details

We have used MIRACL VC1 datasets in which we considered, 15 speakers out of which 10 are female speakers and remaining 5 are male speakers. Each of the speakers spell 10 words 10 times and hence the total dataset we have used becomes 1500.The table 4.1 gives the details of the dataset we used in the project.

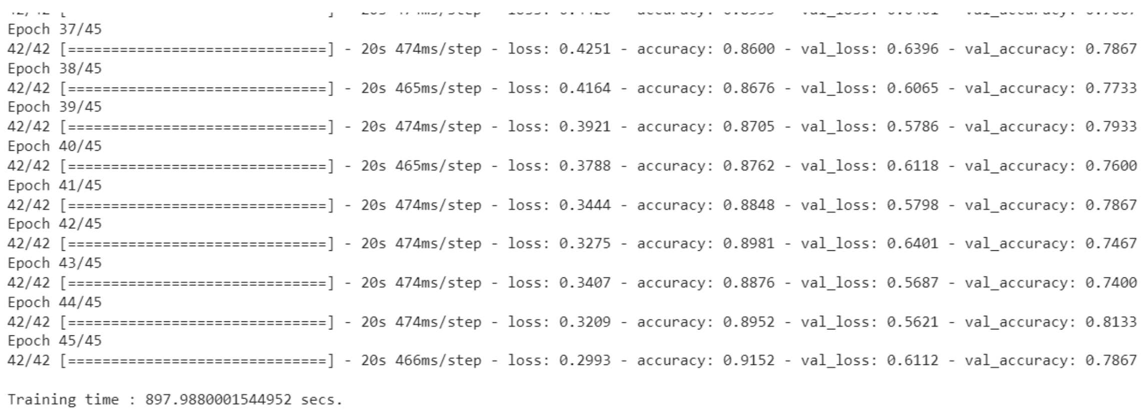
Words	Number of Speakers	Total no. of words Repeated	Total no. of Samples
Begin	15 (10 F + 5M)	10	15*10=150
Choose	15 (10 F + 5M)	10	15*10=150
Connection	15 (10 F + 5M)	10	15*10=150
Navigation	15 (10 F + 5M)	10	15*10=150
Next	15 (10 F + 5M)	10	15*10=150
Previous	15 (10 F + 5M)	10	15*10=150
Start	15 (10 F + 5M)	10	15*10=150
Stop	15 (10 F + 5M)	10	15*10=150
Hello	15 (10 F + 5M)	10	15*10=150
Web	15 (10 F + 5M)	10	15*10=150

F-Female; M- Male

Table 4.1: **Dataset Description**

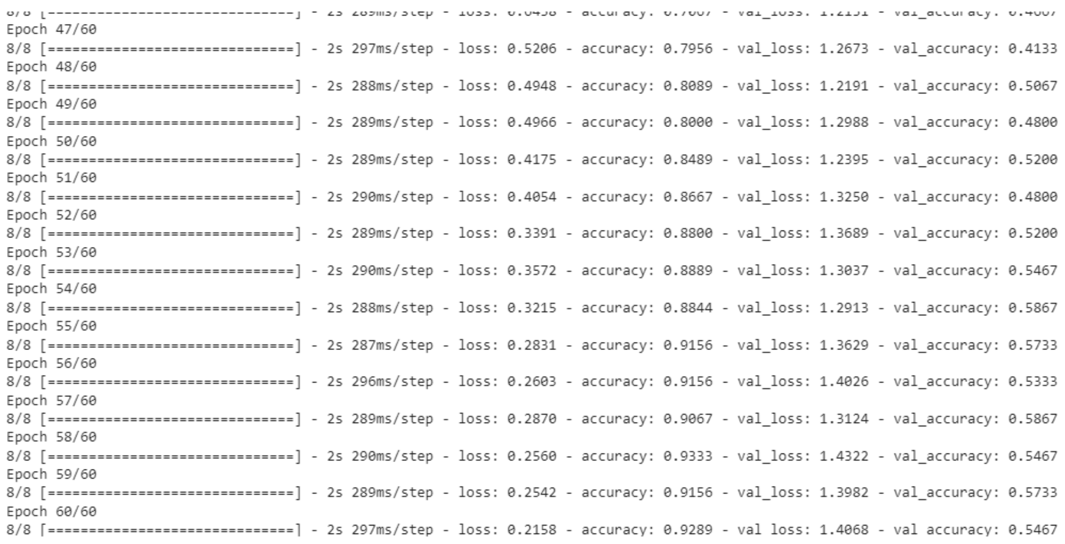
4.1.1 CNN 3D Model

- Dataset is trained for 45 epochs.
- Achieved training accuracy of 91.52%, test accuracy of 78.67% and validation accuracy of 79.00% for English dataset as shown in figure 4.1.
- Achieved training accuracy of 92.89%, test accuracy of 57.33% and validation accuracy of 54.67% for kannada dataset as shown in figure 4.2.



```
Epoch 37/45
42/42 [==============================] - 20s 474ms/step - loss: 0.4251 - accuracy: 0.8600 - val_loss: 0.6396 - val_accuracy: 0.7867
Epoch 38/45
42/42 [==============================] - 20s 465ms/step - loss: 0.4164 - accuracy: 0.8676 - val_loss: 0.6065 - val_accuracy: 0.7733
Epoch 39/45
42/42 [==============================] - 20s 474ms/step - loss: 0.3921 - accuracy: 0.8705 - val_loss: 0.5786 - val_accuracy: 0.7933
Epoch 40/45
42/42 [==============================] - 20s 465ms/step - loss: 0.3788 - accuracy: 0.8762 - val_loss: 0.6118 - val_accuracy: 0.7600
Epoch 41/45
42/42 [==============================] - 20s 474ms/step - loss: 0.3444 - accuracy: 0.8848 - val_loss: 0.5798 - val_accuracy: 0.7867
Epoch 42/45
42/42 [==============================] - 20s 474ms/step - loss: 0.3275 - accuracy: 0.8981 - val_loss: 0.6401 - val_accuracy: 0.7467
Epoch 43/45
42/42 [==============================] - 20s 474ms/step - loss: 0.3407 - accuracy: 0.8876 - val_loss: 0.5687 - val_accuracy: 0.7400
Epoch 44/45
42/42 [==============================] - 20s 474ms/step - loss: 0.3209 - accuracy: 0.8952 - val_loss: 0.5621 - val_accuracy: 0.8133
Epoch 45/45
42/42 [==============================] - 20s 466ms/step - loss: 0.2993 - accuracy: 0.9152 - val_loss: 0.6112 - val_accuracy: 0.7867

Training time : 897.9880001544952 secs.
```

Figure 4.1: **Training English Dataset with CNN 3D model**



```
Epoch 47/60
8/8 [==============================] - 2s 297ms/step - loss: 0.5206 - accuracy: 0.7956 - val_loss: 1.2673 - val_accuracy: 0.4133
Epoch 48/60
8/8 [==============================] - 2s 288ms/step - loss: 0.4948 - accuracy: 0.8089 - val_loss: 1.2191 - val_accuracy: 0.5067
Epoch 49/60
8/8 [==============================] - 2s 289ms/step - loss: 0.4966 - accuracy: 0.8000 - val_loss: 1.2988 - val_accuracy: 0.4800
Epoch 50/60
8/8 [==============================] - 2s 289ms/step - loss: 0.4175 - accuracy: 0.8489 - val_loss: 1.2395 - val_accuracy: 0.5200
Epoch 51/60
8/8 [==============================] - 2s 290ms/step - loss: 0.4054 - accuracy: 0.8667 - val_loss: 1.3250 - val_accuracy: 0.4800
Epoch 52/60
8/8 [==============================] - 2s 289ms/step - loss: 0.3391 - accuracy: 0.8800 - val_loss: 1.3689 - val_accuracy: 0.5200
Epoch 53/60
8/8 [==============================] - 2s 290ms/step - loss: 0.3572 - accuracy: 0.8889 - val_loss: 1.3037 - val_accuracy: 0.5467
Epoch 54/60
8/8 [==============================] - 2s 288ms/step - loss: 0.3215 - accuracy: 0.8844 - val_loss: 1.2913 - val_accuracy: 0.5867
Epoch 55/60
8/8 [==============================] - 2s 287ms/step - loss: 0.2831 - accuracy: 0.9156 - val_loss: 1.3629 - val_accuracy: 0.5733
Epoch 56/60
8/8 [==============================] - 2s 296ms/step - loss: 0.2603 - accuracy: 0.9156 - val_loss: 1.4026 - val_accuracy: 0.5333
Epoch 57/60
8/8 [==============================] - 2s 289ms/step - loss: 0.2870 - accuracy: 0.9067 - val_loss: 1.3124 - val_accuracy: 0.5867
Epoch 58/60
8/8 [==============================] - 2s 290ms/step - loss: 0.2560 - accuracy: 0.9333 - val_loss: 1.4322 - val_accuracy: 0.5467
Epoch 59/60
8/8 [==============================] - 2s 289ms/step - loss: 0.2542 - accuracy: 0.9156 - val_loss: 1.3982 - val_accuracy: 0.5733
Epoch 60/60
8/8 [==============================] - 2s 297ms/step - loss: 0.2158 - accuracy: 0.9289 - val_loss: 1.4068 - val_accuracy: 0.5467
```

Figure 4.2: **Training Kannada Dataset with CNN 3D model**

- Model accuracy and model loss plots of the English datasets are as shown in the Figure 4.3 and Figure 4.4 respectively

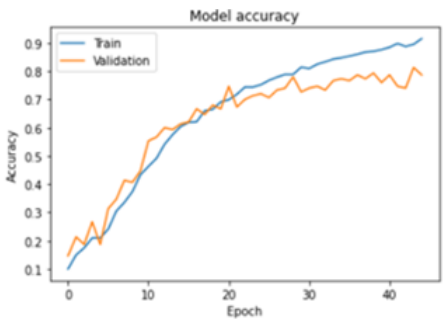


Figure 4.3: **Accuracy plot for CNN 3D model with English Dataset**

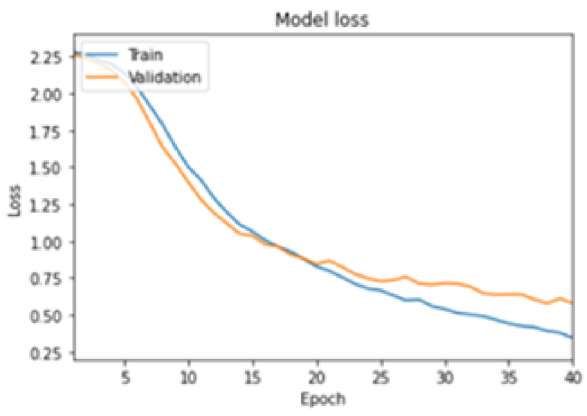


Figure 4.4: **Loss plot for CNN 3D model with English Dataset**

- Model accuracy and model loss plots of the Kannada datasets are as shown in the Figure 4.5 and Figure 4.6 respectively.

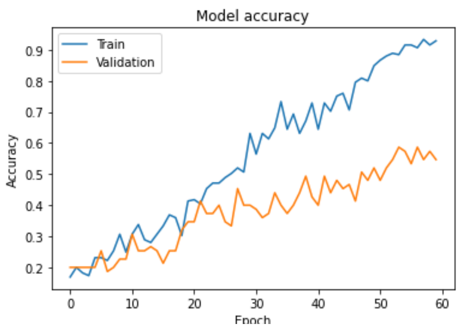


Figure 4.5: **Accuracy plot for CNN 3D model with kannada Dataset**

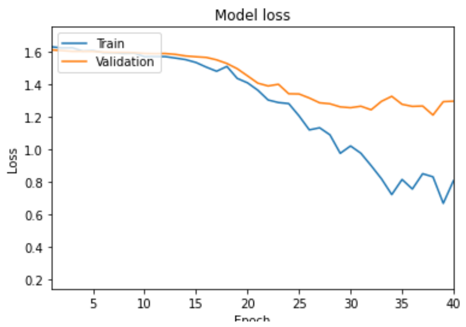


Figure 4.6: **Loss plot for CNN 3D modelwith kannada Dataset**

Confusion matrix indicates the number of words that are being classified as various classes that are available. The confusion matrix provides the accuracy of each word used in the dataset. The accuracy percentage is calculated using the formula,

$$Accuracy(\%) = \frac{w}{W} * 100 \tag{29}$$

where,

w- total number of correctly predicted instances for a word in the dataset

W- total number of instances for a word used in dataset

Total number of instances for each word is 30. The confusion matrix of the CNN 3D model which is used to train English datasets is shown in Figure 4.4. From the figure 4.4 which represents the confusion matrix, it is observed that for a word classified as “Begin” in the test dataset which belongs to “Begin instances”, 21 instances are predicted correctly while 9 others are predicted incorrectly. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{21}{30} * 100 = 70.00\%$$

Similarly for a word classified as “Choose” in the test dataset which belongs to “Choose instances”, 29 instances are predicted correctly while 1 instance is incorrectly predicted. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{29}{30} * 100 = 96.67\%$$

Similarly for a word classified as “Connection” in the test dataset which belongs to “Connection instances”, 24 instances are predicted correctly while 6 other instances are incorrectly predicted. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{24}{30} * 100 = 80.00\%$$

Similarly for a word classified as “Navigation” in the test dataset which belongs to “Navigation instances”, 21 instances are predicted correctly while 9 other instances are incorrectly predicted. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{21}{30} * 100 = 70.00\%$$

Similarly for a word classified as “Next” in the test dataset which belongs to “Next instances”, 24 instances are predicted correctly while 6 other instances are incorrectly predicted. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{24}{30} * 100 = 80.00\%$$

Similarly for a word classified as “Previous” in the test dataset which belongs to “Previous instances”, 23 instances are predicted correctly while 7 other instances are incorrectly predicted. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{23}{30} * 100 = 76.67\%$$

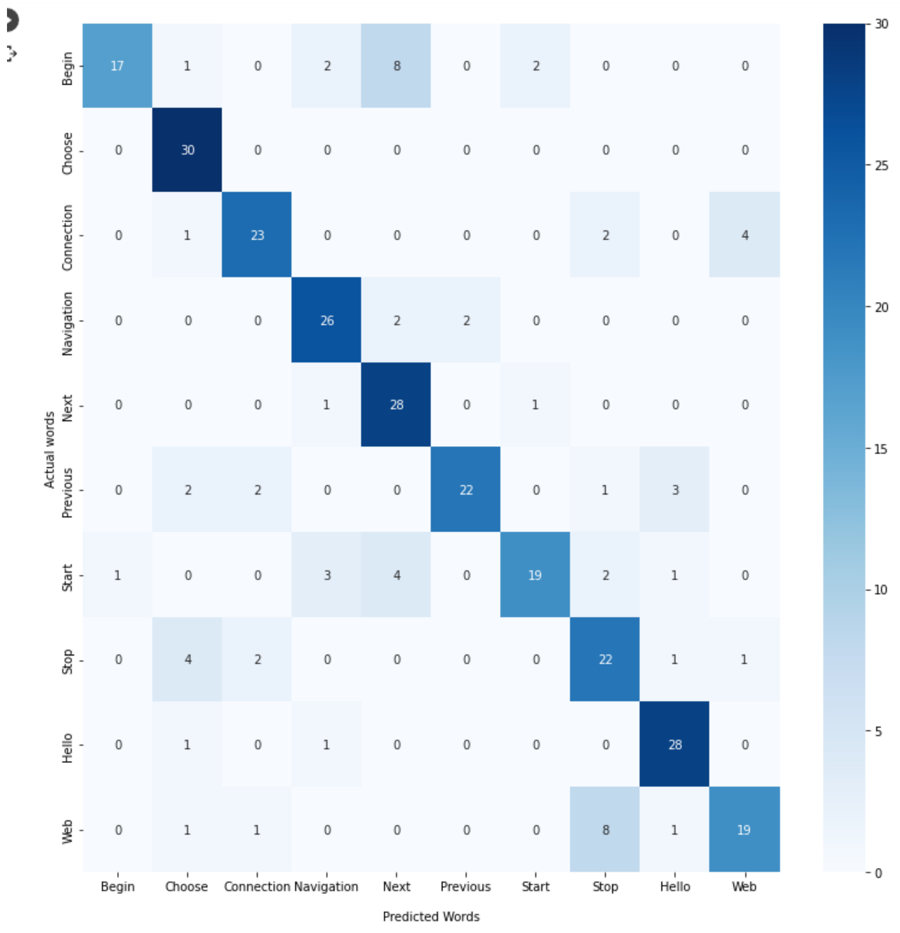


Figure 4.7: **Confusion matrix for CNN 3D model with English Dataset**

Similarly for a word classified as “Start” in the test dataset which belongs to “Start instances”, 26 instances are predicted correctly while 4 other instances are incorrectly predicted. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{26}{30} * 100 = 86.67\%$$

Similarly for a word classified as “Stop” in the test dataset which belongs to “Stop instances”, 25 instances are predicted correctly while 5 other instances are incorrectly predicted. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{25}{30} * 100 = 83.33\%$$

Similarly for a word classified as “Hello” in the test dataset which belongs to “Hello instances”, 25 instances are predicted correctly while 5 other instances are incorrectly predicted. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{25}{30} * 100 = 83.33\%$$

Similarly for a word classified as “Web” in the test dataset which belongs to “Web instances”, 18 instances are predicted correctly while 12 other instances are incorrectly predicted. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{18}{30} * 100 = 60.00\%$$

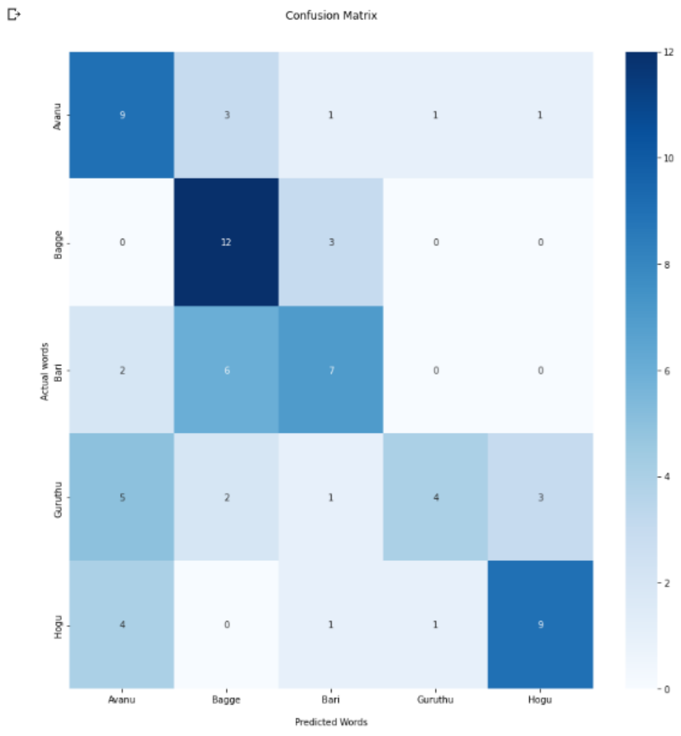


Figure 4.8: **Confusion matrix for CNN 3D model with kannada Dataset**

From the Figure 4.8, The confusion matrix for Kannada datasets can be obtained. It is observed that for a word classified as “Avanu” in the test dataset which belongs to “Avanu instances”, 9 instances are predicted correctly while 6 others are predicted incorrectly. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{9}{15} * 100 = 60.00\%$$

Similarly for a word classified as “Bagge” in the test dataset which belongs to “Bagge instances”, 12 instances are predicted correctly while 3 instance is incorrectly predicted.

Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{12}{15} * 100 = 80.00\%$$

Similarly for a word classified as “Bari” in the test dataset which belongs to “Bari instances”, 7 instances are predicted correctly while 8 other instances are incorrectly predicted. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{7}{15} * 100 = 46.67\%$$

Similarly for a word classified as “Guruthu” in the test dataset which belongs to “Guruthu instances”, 4 instances are predicted correctly while 11 other instances are incorrectly predicted. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{4}{15} * 100 = 26.67\%$$

Similarly for a word classified as “Hogu” in the test dataset which belongs to “Hogu instances”, 9 instances are predicted correctly while 6 other instances are incorrectly predicted. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{9}{15} * 100 = 60.00\%$$

Classification report gives Precision, Recall and F1 score for every classification where

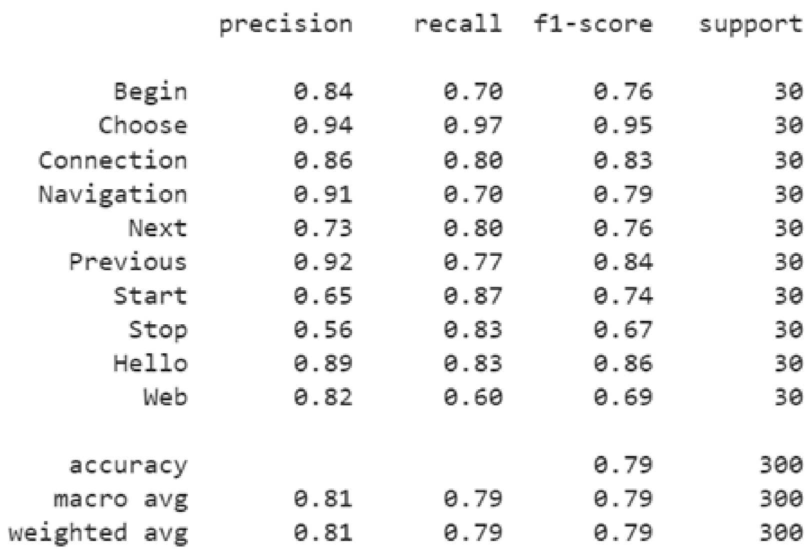
$$Precision = \frac{TruePositive}{(TruePositive + FalsePositive)} \tag{30}$$

$$Recall = \frac{TruePositive}{(TruePositive + FalseNegative)} \tag{31}$$

$$f1score = \frac{2 * (Recall * Precision)}{(Recall + Precision)} \tag{32}$$

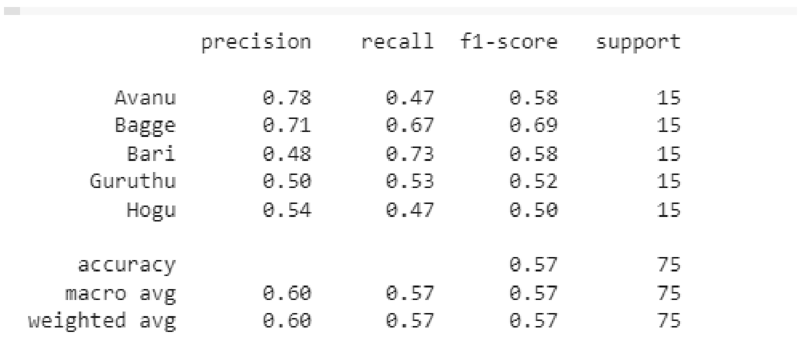
The classification matrix of English dataset is shown below in the Figure 4.9.

The classification matrix of Kannada dataset is shown below in the Figure 4.10.



```
              precision    recall  f1-score   support

       Begin       0.84      0.70      0.76        30
      Choose       0.94      0.97      0.95        30
  Connection       0.86      0.80      0.83        30
  Navigation       0.91      0.70      0.79        30
        Next       0.73      0.80      0.76        30
    Previous       0.92      0.77      0.84        30
       Start       0.65      0.87      0.74        30
        Stop       0.56      0.83      0.67        30
       Hello       0.89      0.83      0.86        30
         Web       0.82      0.60      0.69        30

    accuracy                           0.79       300
   macro avg       0.81      0.79      0.79       300
weighted avg       0.81      0.79      0.79       300
```

Figure 4.9: **Classification matrix for CNN 3D model with English Dataset**

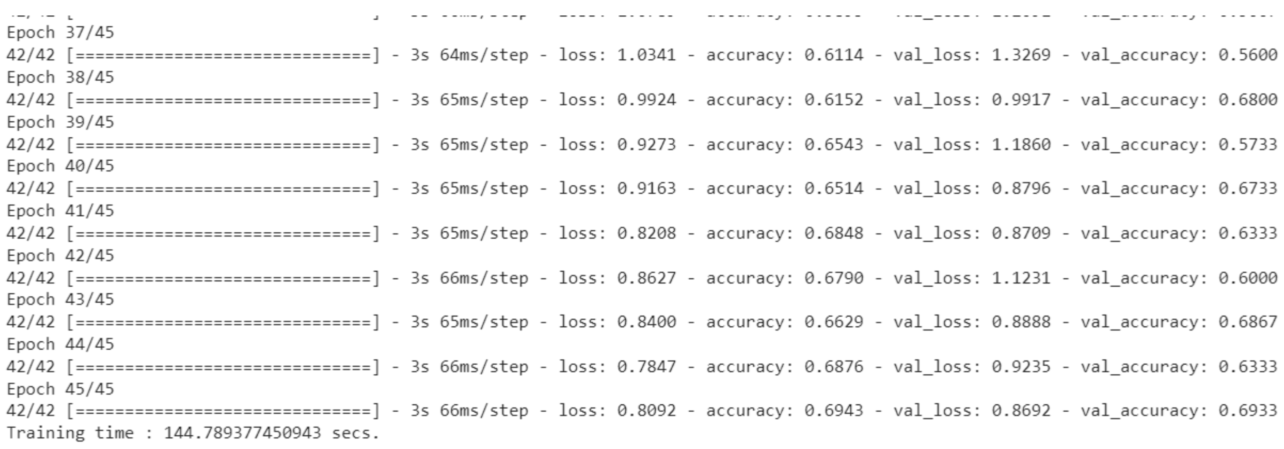


```
              precision    recall  f1-score   support

       Avanu       0.78      0.47      0.58        15
       Bagge       0.71      0.67      0.69        15
        Bari       0.48      0.73      0.58        15
     Guruthu       0.50      0.53      0.52        15
        Hogu       0.54      0.47      0.50        15

    accuracy                           0.57        75
   macro avg       0.60      0.57      0.57        75
weighted avg       0.60      0.57      0.57        75
```

Figure 4.10: **Classification matrix for CNN 3D model with Kannada Dataset**

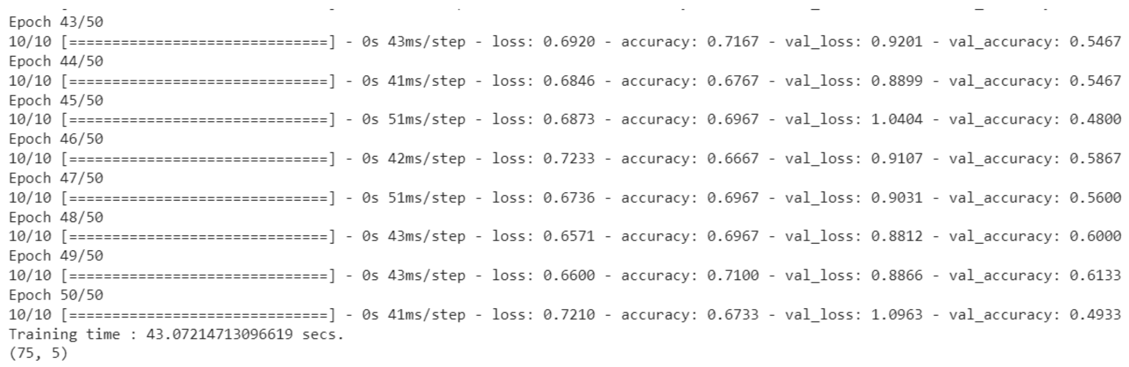
4.2 Result For Visual Speech Recognition Using LSTM Model

- Dataset is trained for 45 epochs.
- Achieved training accuracy of 69.43% , test accuracy of 66.00% and validation accuracy of 69.33% for English dataset as shown in figure 4.11
- Achieved training accuracy of 67.33% , test accuracy of 49.00% and validation accuracy of 49.33% for English dataset as shown in figure 4.12
- Model accuracy and model loss plots for English Dataset are shown in the below Figure 4.13 and Figure 4.14



```
Epoch 37/45
42/42 [==============================] - 3s 64ms/step - loss: 1.0341 - accuracy: 0.6114 - val_loss: 1.3269 - val_accuracy: 0.5600
Epoch 38/45
42/42 [==============================] - 3s 65ms/step - loss: 0.9924 - accuracy: 0.6152 - val_loss: 0.9917 - val_accuracy: 0.6800
Epoch 39/45
42/42 [==============================] - 3s 65ms/step - loss: 0.9273 - accuracy: 0.6543 - val_loss: 1.1860 - val_accuracy: 0.5733
Epoch 40/45
42/42 [==============================] - 3s 65ms/step - loss: 0.9163 - accuracy: 0.6514 - val_loss: 0.8796 - val_accuracy: 0.6733
Epoch 41/45
42/42 [==============================] - 3s 65ms/step - loss: 0.8208 - accuracy: 0.6848 - val_loss: 0.8709 - val_accuracy: 0.6333
Epoch 42/45
42/42 [==============================] - 3s 66ms/step - loss: 0.8627 - accuracy: 0.6790 - val_loss: 1.1231 - val_accuracy: 0.6000
Epoch 43/45
42/42 [==============================] - 3s 65ms/step - loss: 0.8400 - accuracy: 0.6629 - val_loss: 0.8888 - val_accuracy: 0.6867
Epoch 44/45
42/42 [==============================] - 3s 66ms/step - loss: 0.7847 - accuracy: 0.6876 - val_loss: 0.9235 - val_accuracy: 0.6333
Epoch 45/45
42/42 [==============================] - 3s 66ms/step - loss: 0.8092 - accuracy: 0.6943 - val_loss: 0.8692 - val_accuracy: 0.6933
Training time : 144.789377450943 secs.
```

Figure 4.11: **Training English Dataset with LSTM model.**



```
Epoch 43/50
10/10 [==============================] - 0s 43ms/step - loss: 0.6920 - accuracy: 0.7167 - val_loss: 0.9201 - val_accuracy: 0.5467
Epoch 44/50
10/10 [==============================] - 0s 41ms/step - loss: 0.6846 - accuracy: 0.6767 - val_loss: 0.8899 - val_accuracy: 0.5467
Epoch 45/50
10/10 [==============================] - 0s 51ms/step - loss: 0.6873 - accuracy: 0.6967 - val_loss: 1.0404 - val_accuracy: 0.4800
Epoch 46/50
10/10 [==============================] - 0s 42ms/step - loss: 0.7233 - accuracy: 0.6667 - val_loss: 0.9107 - val_accuracy: 0.5867
Epoch 47/50
10/10 [==============================] - 0s 51ms/step - loss: 0.6736 - accuracy: 0.6967 - val_loss: 0.9031 - val_accuracy: 0.5600
Epoch 48/50
10/10 [==============================] - 0s 43ms/step - loss: 0.6571 - accuracy: 0.6967 - val_loss: 0.8812 - val_accuracy: 0.6000
Epoch 49/50
10/10 [==============================] - 0s 43ms/step - loss: 0.6600 - accuracy: 0.7100 - val_loss: 0.8866 - val_accuracy: 0.6133
Epoch 50/50
10/10 [==============================] - 0s 41ms/step - loss: 0.7210 - accuracy: 0.6733 - val_loss: 1.0963 - val_accuracy: 0.4933
Training time : 43.07214713096619 secs.
(75, 5)
```

Figure 4.12: **Training Kannada Dataset with LSTM model.**

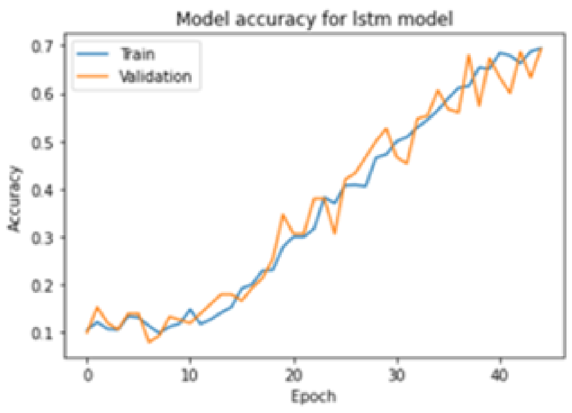


Figure 4.13: **Accuracy plot with English dataset with LSTM model.**

- Model accuracy and model loss plots for Kannada Dataset are shown in the below Figure 4.15 and Figure 4.16

Confusion matrix indicates the number of words that are being classified as various classes that are available. The confusion matrix provides the accuracy of each word used in the

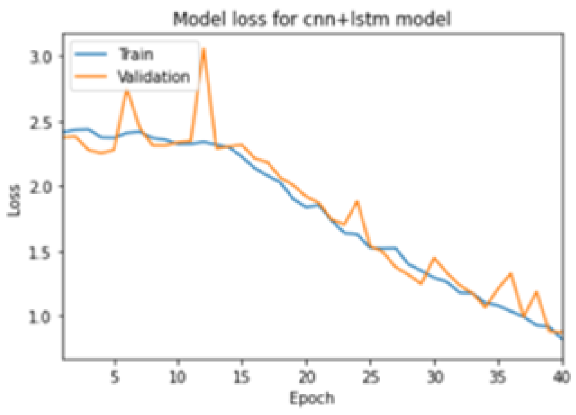


Figure 4.14: **Loss plot with English dataset with LSTM model.**

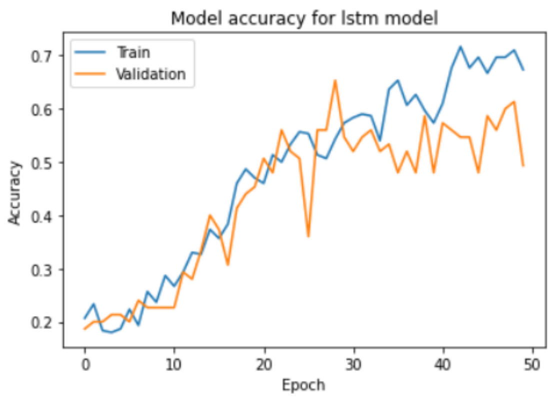


Figure 4.15: **Accuracy plot with Kannada dataset with LSTM model.**

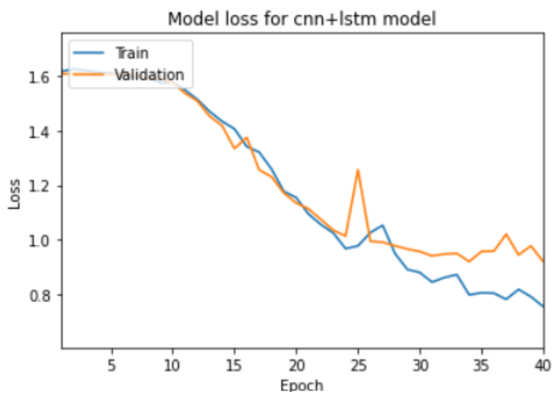


Figure 4.16: **Loss plot with Kannada dataset with LSTM model.**

dataset. The accuracy percentage is calculated using the formula,

$$Accuracy(\%) = \frac{w}{W} * 100 \tag{33}$$

where,

w- total number of correctly predicted instances for a word in the dataset
W- total number of instances for a word used in dataset
The confusion matrix of the hybrid model (CNN 3D + LSTM) which is used to train English datasets is shown in Figure 4.17. From the figure 4.17 which represents the confusion matrix it is observed that for a word classified as "Begin" in the test dataset which belongs to "Begin instances", 15 instances are predicted correctly while 15 others are predicted incorrectly. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{15}{30} * 100 = 50.00\%$$

Similarly for a word classified as "Choose" in the test dataset which belongs to "Choose instances", 26 instances are predicted correctly while 4 instances are incorrectly predicted. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{26}{30} * 100 = 86.67\%$$

Similarly for a word classified as "Connection" in the test dataset which belongs to "Connection instances", 27 instances are predicted correctly while 3 other instances are incorrectly predicted. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{27}{30} * 100 = 90.00\%$$

Similarly for a word classified as "Navigation" in the test dataset which belongs to "Navigation instances", 27 instances are predicted correctly while 3 other instances are incorrectly predicted. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{27}{30} * 100 = 90.00\%$$

Similarly for a word classified as "Next" in the test dataset which belongs to "Next instances", 21 instances are predicted correctly while 9 other instances are incorrectly predicted. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{21}{30} * 100 = 70.00\%$$

Similarly for a word classified as "Previous" in the test dataset which belongs to "Previous instances", 11 instances are predicted correctly while 19 other instances are incorrectly predicted. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{11}{30} * 100 = 36.67\%$$

Similarly for a word classified as “Start” in the test dataset which belongs to “Start instances”, 21 instances are predicted correctly while 9 other instances are incorrectly predicted. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{21}{30} * 100 = 70.00\%$$

Similarly for a word classified as “Stop” in the test dataset which belongs to “Stop in-

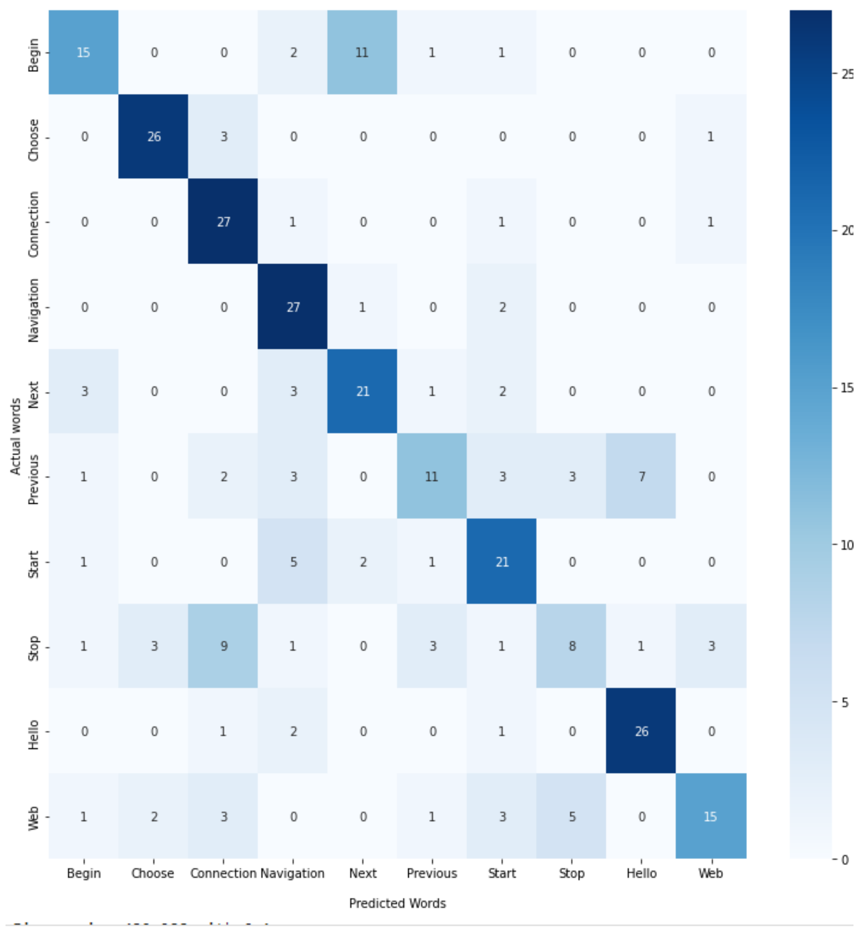


Figure 4.17: **Confusion matrix with LSTM model for English Dataset**

stances”, 8 instances are predicted correctly while 22 other instances are incorrectly predicted. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{8}{30} * 100 = 26.67\%$$

Similarly for a word classified as “Hello” in the test dataset which belongs to “Hello instances”, 26 instances are predicted correctly while 4 other instances are incorrectly predicted. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{26}{30} * 100 = 86.67\%$$

Similarly for a word classified as “Web” in the test dataset which belongs to “Web instances”, 15 instances are predicted correctly while 15 other instances are incorrectly predicted. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{15}{30} * 100 = 50.00\%$$

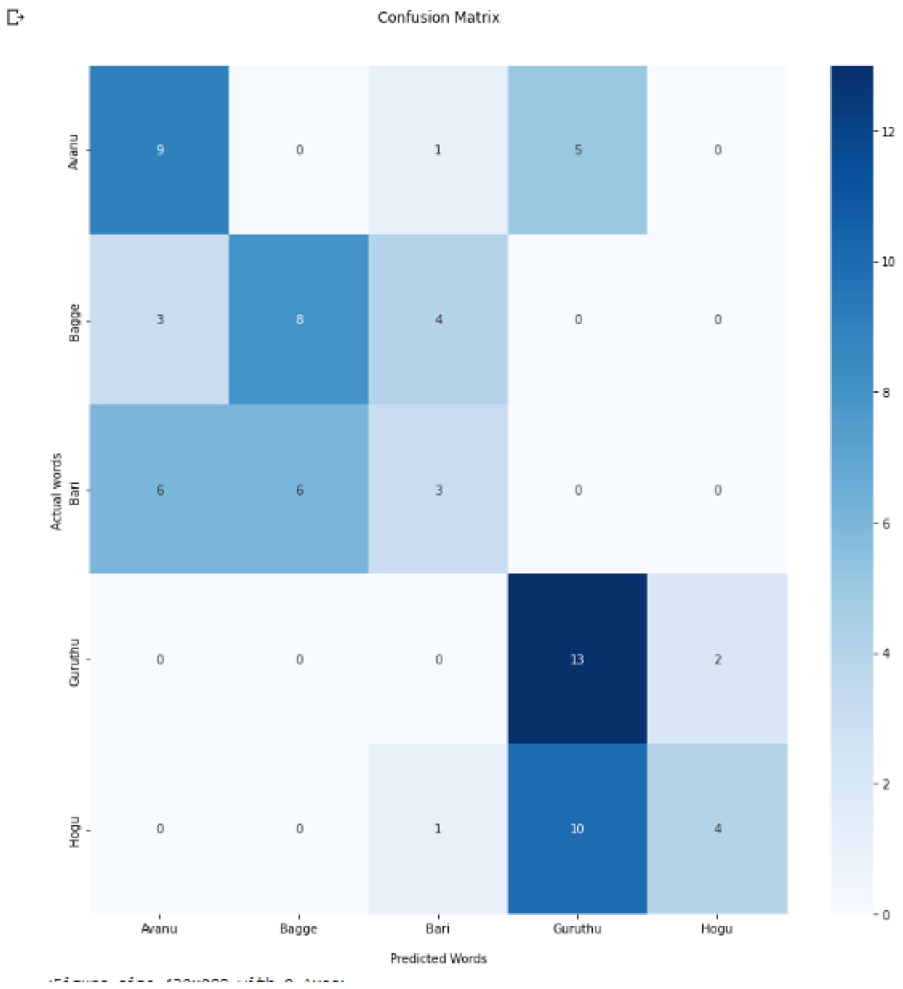


Figure 4.18: **Confusion matrix for LSTM model for Kannada Dataset**

From the Figure 4.18, The confusion matrix for Kannada datasets can be obtained. It is observed that for a word classified as “Avanu” in the test dataset which belongs to “Avanu instances”, 9 instances are predicted correctly while 6 others are predicted incorrectly. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{9}{15} * 100 = 60.00\%$$

Similarly for a word classified as “Bagge” in the test dataset which belongs to “Bagge instances”, 8 instances are predicted correctly while 7 instance is incorrectly predicted.

Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{8}{15} * 100 = 53.33\%$$

Similarly for a word classified as “Bari” in the test dataset which belongs to “Bari instances”, 3 instances are predicted correctly while 12 other instances are incorrectly predicted. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{3}{15} * 100 = 20.00\%$$

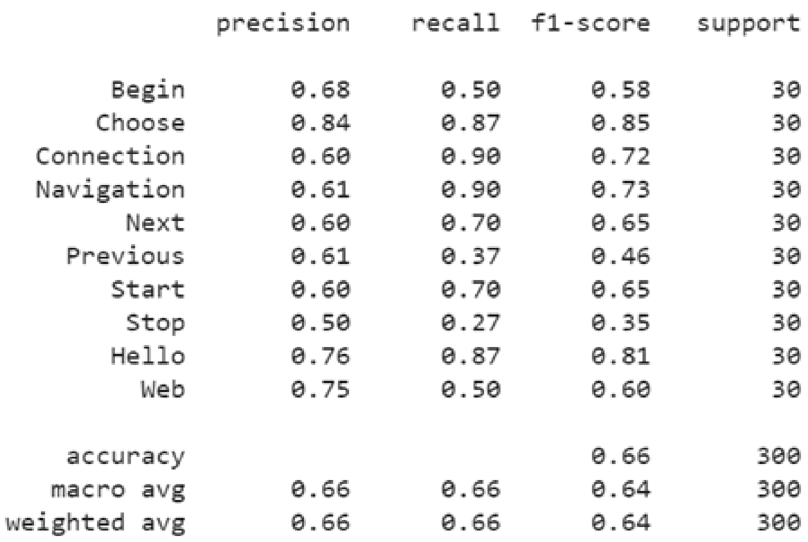
Similarly for a word classified as “Guruthu” in the test dataset which belongs to “Guruthu instances”, 13 instances are predicted correctly while 2 other instances are incorrectly predicted. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{13}{15} * 100 = 86.67\%$$

Similarly for a word classified as “Hogu” in the test dataset which belongs to “Hogu instances”, 4 instances are predicted correctly while 11 other instances are incorrectly predicted. Hence the accuracy percentage is given as,

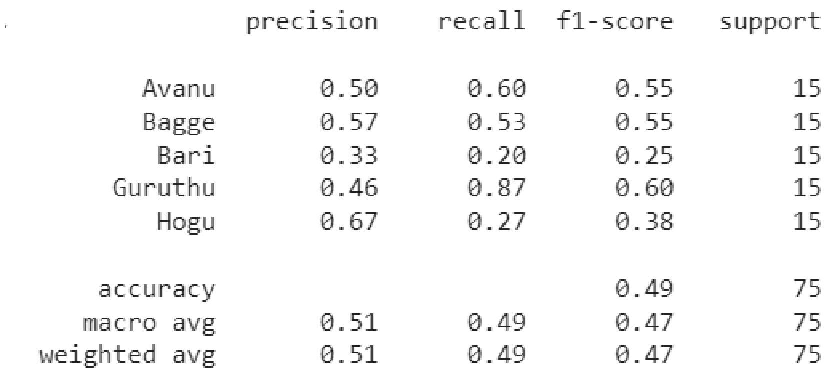
$$Accuracy(\%) = \frac{4}{15} * 100 = 26.67\%$$

The classification matrix of English dataset is shown below in the Figure 4.19 and for Kannada Dataset is shown in Figure 4.20



```
              precision    recall  f1-score   support

       Begin       0.68      0.50      0.58        30
      Choose       0.84      0.87      0.85        30
  Connection       0.60      0.90      0.72        30
  Navigation       0.61      0.90      0.73        30
        Next       0.60      0.70      0.65        30
    Previous       0.61      0.37      0.46        30
       Start       0.60      0.70      0.65        30
        Stop       0.50      0.27      0.35        30
       Hello       0.76      0.87      0.81        30
         Web       0.75      0.50      0.60        30

    accuracy                           0.66       300
   macro avg       0.66      0.66      0.64       300
weighted avg       0.66      0.66      0.64       300
```

Figure 4.19: **Classification matrix with LSTM model for English Dataset**

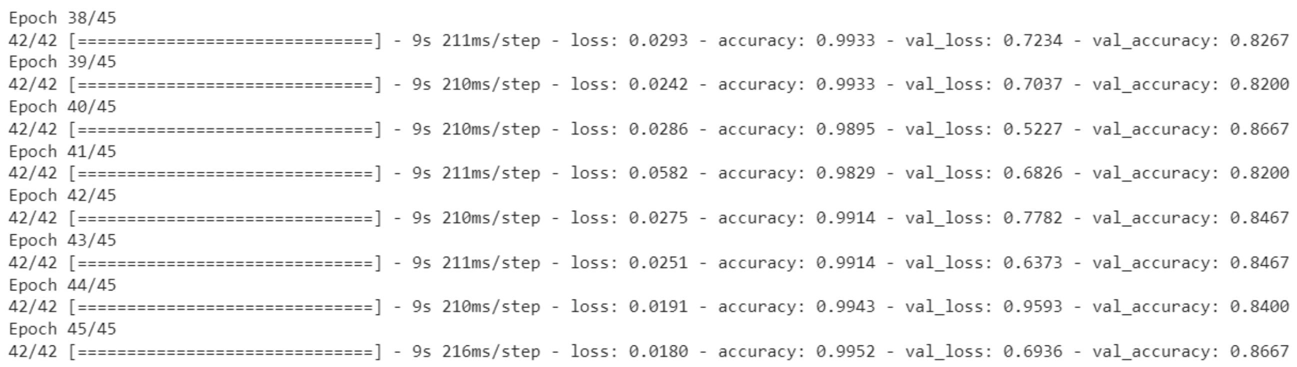


```
              precision    recall  f1-score   support

       Avanu       0.50      0.60      0.55        15
       Bagge       0.57      0.53      0.55        15
        Bari       0.33      0.20      0.25        15
     Guruthu       0.46      0.87      0.60        15
        Hogu       0.67      0.27      0.38        15

    accuracy                           0.49        75
   macro avg       0.51      0.49      0.47        75
weighted avg       0.51      0.49      0.47        75
```

Figure 4.20: **Classification matrix with LSTM model for Kannada Dataset**

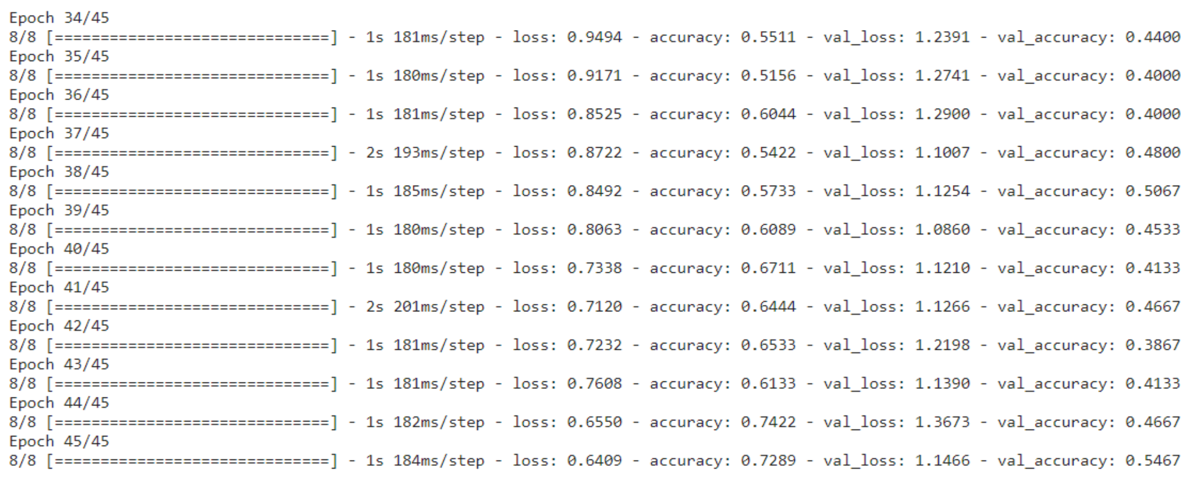
4.3 Result for Visual Speech Recognition Using Hybrid Model

- Dataset is trained for 45 epochs.
- Achieved training accuracy of 99.52% , test accuracy of 85.00% and validation accuracy of 86.67% for English dataset as shown in figure 4.21.
- Achieved training accuracy of 72.89%, test accuracy of 61.00% and validation accuracy of 54.67% for Kannada dataset as shown in figure 4.22.



```
Epoch 38/45
42/42 [==============================] - 9s 211ms/step - loss: 0.0293 - accuracy: 0.9933 - val_loss: 0.7234 - val_accuracy: 0.8267
Epoch 39/45
42/42 [==============================] - 9s 210ms/step - loss: 0.0242 - accuracy: 0.9933 - val_loss: 0.7037 - val_accuracy: 0.8200
Epoch 40/45
42/42 [==============================] - 9s 210ms/step - loss: 0.0286 - accuracy: 0.9895 - val_loss: 0.5227 - val_accuracy: 0.8667
Epoch 41/45
42/42 [==============================] - 9s 211ms/step - loss: 0.0582 - accuracy: 0.9829 - val_loss: 0.6826 - val_accuracy: 0.8200
Epoch 42/45
42/42 [==============================] - 9s 210ms/step - loss: 0.0275 - accuracy: 0.9914 - val_loss: 0.7782 - val_accuracy: 0.8467
Epoch 43/45
42/42 [==============================] - 9s 211ms/step - loss: 0.0251 - accuracy: 0.9914 - val_loss: 0.6373 - val_accuracy: 0.8467
Epoch 44/45
42/42 [==============================] - 9s 210ms/step - loss: 0.0191 - accuracy: 0.9943 - val_loss: 0.9593 - val_accuracy: 0.8400
Epoch 45/45
42/42 [==============================] - 9s 216ms/step - loss: 0.0180 - accuracy: 0.9952 - val_loss: 0.6936 - val_accuracy: 0.8667
```

Figure 4.21: **Training dataset with hybrid model for English Dataset**



```
Epoch 34/45
8/8 [==============================] - 1s 181ms/step - loss: 0.9494 - accuracy: 0.5511 - val_loss: 1.2391 - val_accuracy: 0.4400
Epoch 35/45
8/8 [==============================] - 1s 180ms/step - loss: 0.9171 - accuracy: 0.5156 - val_loss: 1.2741 - val_accuracy: 0.4000
Epoch 36/45
8/8 [==============================] - 1s 181ms/step - loss: 0.8525 - accuracy: 0.6044 - val_loss: 1.2900 - val_accuracy: 0.4000
Epoch 37/45
8/8 [==============================] - 2s 193ms/step - loss: 0.8722 - accuracy: 0.5422 - val_loss: 1.1007 - val_accuracy: 0.4800
Epoch 38/45
8/8 [==============================] - 1s 185ms/step - loss: 0.8492 - accuracy: 0.5733 - val_loss: 1.1254 - val_accuracy: 0.5067
Epoch 39/45
8/8 [==============================] - 1s 180ms/step - loss: 0.8063 - accuracy: 0.6089 - val_loss: 1.0860 - val_accuracy: 0.4533
Epoch 40/45
8/8 [==============================] - 1s 180ms/step - loss: 0.7338 - accuracy: 0.6711 - val_loss: 1.1210 - val_accuracy: 0.4133
Epoch 41/45
8/8 [==============================] - 2s 201ms/step - loss: 0.7120 - accuracy: 0.6444 - val_loss: 1.1266 - val_accuracy: 0.4667
Epoch 42/45
8/8 [==============================] - 1s 181ms/step - loss: 0.7232 - accuracy: 0.6533 - val_loss: 1.2198 - val_accuracy: 0.3867
Epoch 43/45
8/8 [==============================] - 1s 181ms/step - loss: 0.7608 - accuracy: 0.6133 - val_loss: 1.1390 - val_accuracy: 0.4133
Epoch 44/45
8/8 [==============================] - 1s 182ms/step - loss: 0.6550 - accuracy: 0.7422 - val_loss: 1.3673 - val_accuracy: 0.4667
Epoch 45/45
8/8 [==============================] - 1s 184ms/step - loss: 0.6409 - accuracy: 0.7289 - val_loss: 1.1466 - val_accuracy: 0.5467
```

Figure 4.22: **Training dataset with hybrid model for Kannada Dataset**

- Model accuracy and model loss plots for English dataset are shown in the below Figure 4.23 and Figure 4.24

- Model accuracy and model loss plots for Kannada dataset are shown in the below Figure 4.25 and Figure 4.26

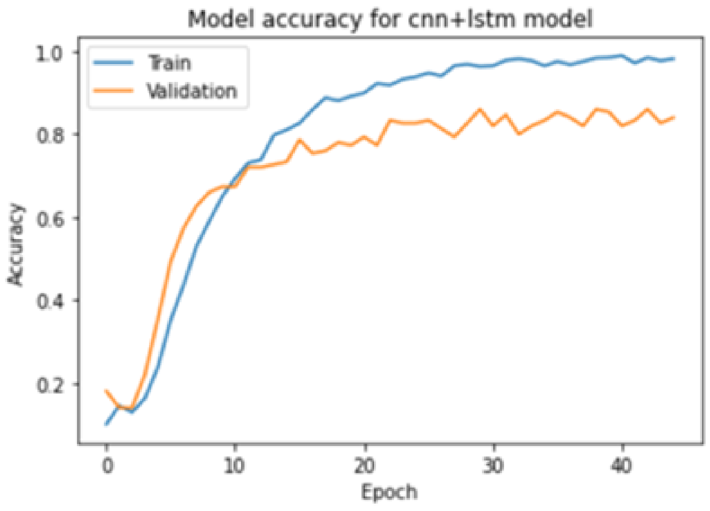


Figure 4.23: **Accuracy plot with hybrid model with English Dataset**

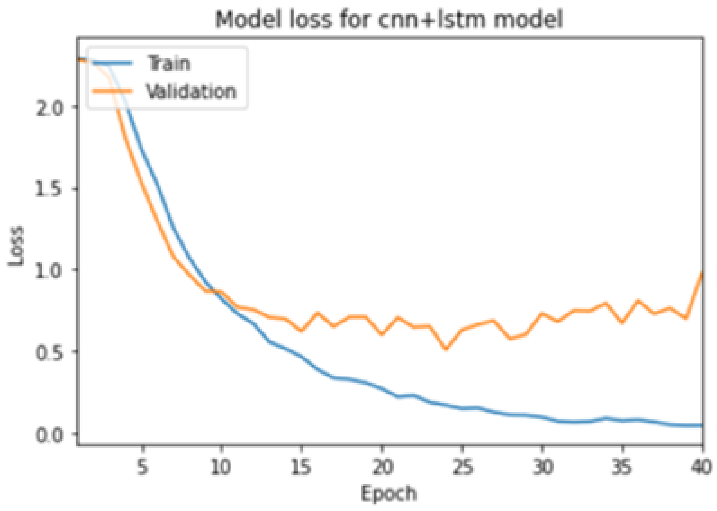


Figure 4.24: **Loss plot with hybrid model with English Dataset**

Confusion matrix indicates the number of words that are being classified as various classes that are available. The confusion matrix provides the accuracy of each word used in the dataset. The accuracy percentage is calculated using the formula,

$$Accuracy(\%) = \frac{w}{W} * 100 \tag{34}$$

where,

w- total number of correctly predicted instances for a word in the dataset

W- total number of instances for a word used in dataset

The confusion matrix of the hybrid model (CNN 3D + LSTM) which is used to train English datasets is shown in Figure 4.27. From the figure 4.27 which represents the confusion

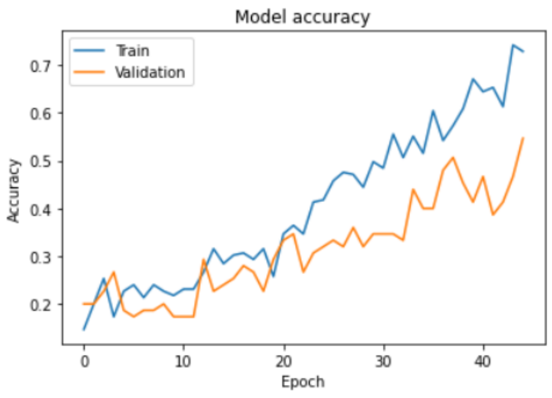


Figure 4.25: **Accuracy plot with hybrid model for Kannada Dataset**

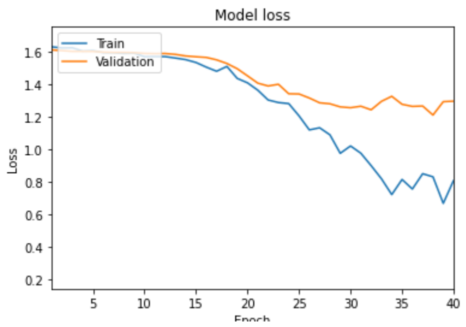


Figure 4.26: **Loss plot with hybrid model for Kannada Dataset**

matrix it is observed that for a word classified as "Begin" in the test dataset which belongs to "Begin instances", 25 instances are predicted correctly while 5 others are predicted incorrectly. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{25}{30} * 100 = 83.33\%$$

Similarly for a word classified as "Choose" in the test dataset which belongs to "Choose instances", 27 instances are predicted correctly while 3 instances are incorrectly predicted. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{27}{30} * 100 = 90.00\%$$

Similarly for a word classified as "Connection" in the test dataset which belongs to "Connection instances", 21 instances are predicted correctly while 9 other instances are incorrectly predicted. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{21}{30} * 100 = 70.00\%$$

Similarly for a word classified as “Navigation” in the test dataset which belongs to “Navigation instances”, 24 instances are predicted correctly while 6 other instances are incorrectly predicted. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{24}{30} * 100 = 80.00\%$$

Similarly for a word classified as “Next” in the test dataset which belongs to “Next instances”, 28 instances are predicted correctly while 2 other instances are incorrectly predicted. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{24}{30} * 100 = 80.00\%$$

Similarly for a word classified as “Previous” in the test dataset which belongs to “Previous instances”, 25 instances are predicted correctly while 5 other instances are incorrectly predicted. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{25}{30} * 100 = 83.33\%$$

Similarly for a word classified as “Start” in the test dataset which belongs to “Start instances”, 25 instances are predicted correctly while 5 other instances are incorrectly predicted. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{25}{30} * 100 = 83.33\%$$

Similarly for a word classified as “Stop” in the test dataset which belongs to “Stop instances”, 25 instances are predicted correctly while 5 other instances are incorrectly predicted. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{25}{30} * 100 = 83.33\%$$

Similarly for a word classified as “Hello” in the test dataset which belongs to “Hello instances”, 26 instances are predicted correctly while 4 other instances are incorrectly predicted. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{26}{30} * 100 = 86.67\%$$

Similarly for a word classified as “Web” in the test dataset which belongs to “Web instances”, 21 instances are predicted correctly while 9 other instances are incorrectly predicted. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{21}{30} * 100 = 70.00\%$$

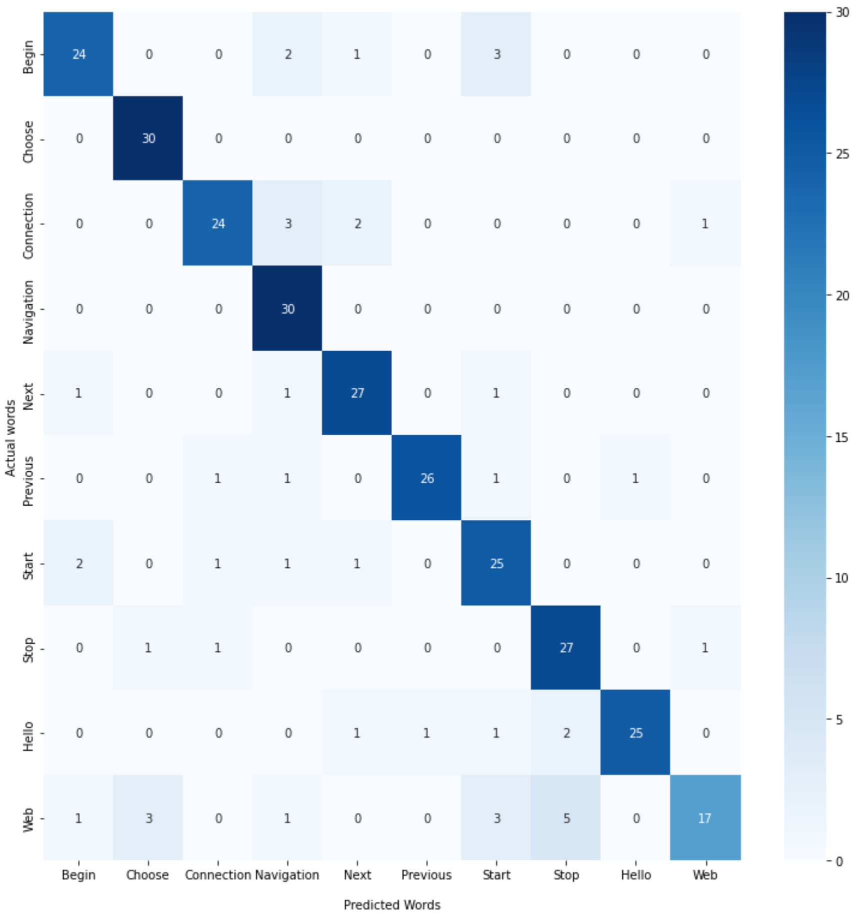


Figure 4.27: **Confusion matrix of the English dataset with hybrid model**

The confusion matrix of the hybrid model (CNN 3D + LSTM) which is used to train Kannada datasets is shown in Figure 4.28. From the figure 4.28 which represents the confusion matrix it is observed that for a word classified as “Avanu” in the test dataset which belongs to “Avanu instances”, 10 instances are predicted correctly while 5 others are predicted incorrectly. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{10}{15} * 100 = 66.67\%$$

Similarly for a word classified as “Bagge” in the test dataset which belongs to “Bagge instances”, 11 instances are predicted correctly while 4 instances are incorrectly predicted. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{11}{15} * 100 = 73.33\%$$

Similarly for a word classified as “Bari” in the test dataset which belongs to “Bari instances”, 9 instances are predicted correctly while 6 other instances are incorrectly predicted. Hence the accuracy percentage is given as,

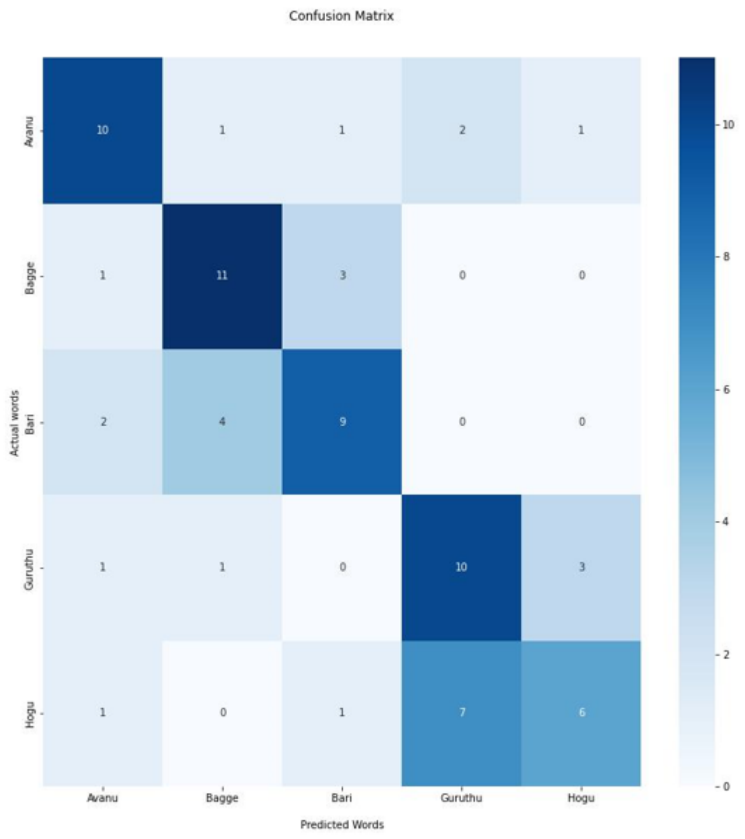
$$Accuracy(\%) = \frac{9}{15} * 100 = 60.00\%$$

Similarly for a word classified as "Guruthu" in the test dataset which belongs to "Guruthu instances", 10 instances are predicted correctly while 5 other instances are incorrectly predicted. Hence the accuracy percentage is given as,

$$Accuracy(\%) = \frac{10}{15} * 100 = 66.67\%$$

Similarly for a word classified as "Hogu" in the test dataset which belongs to "Hogu instances", 6 instances are predicted correctly while 9 other instances are incorrectly predicted. Hence the accuracy percentage is given as,

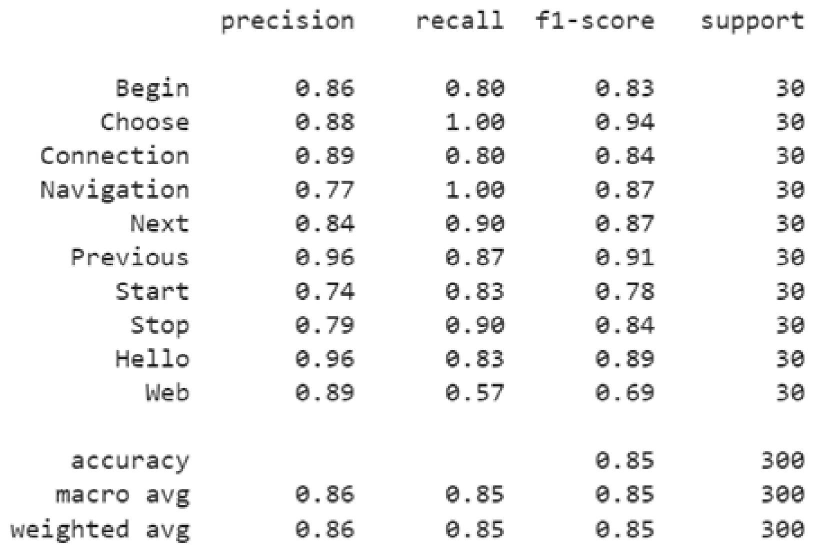
$$Accuracy(\%) = \frac{6}{15} * 100 = 40.00\%$$



	Avanu	Bagge	Bari	Guruthu	Hogu
Avanu	10	1	1	2	1
Bagge	1	11	3	0	0
Bari	2	4	9	0	0
Guruthu	1	1	0	10	3
Hogu	1	0	1	7	6

Figure 4.28: **Confusion matrix of the Kannada dataset with hybrid model**

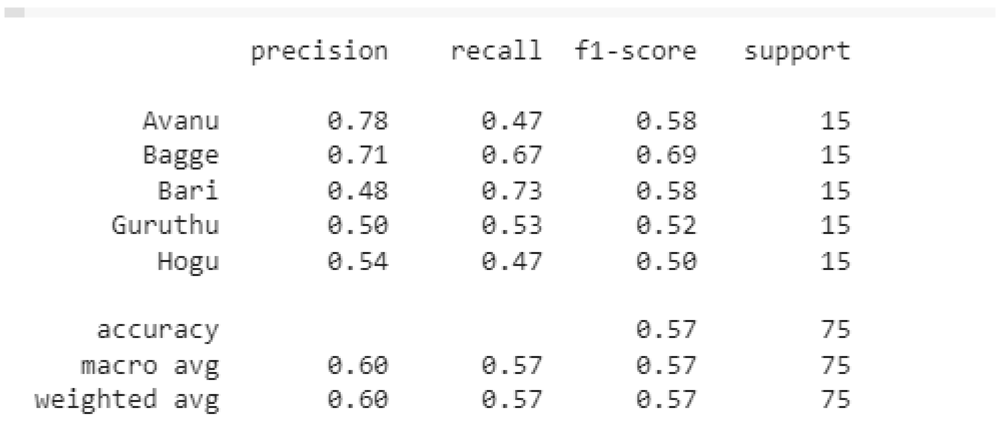
The classification matrix of English dataset is shown below in the Figure 4.29



```
              precision    recall  f1-score   support

       Begin       0.86      0.80      0.83        30
      Choose       0.88      1.00      0.94        30
  Connection       0.89      0.80      0.84        30
  Navigation       0.77      1.00      0.87        30
        Next       0.84      0.90      0.87        30
    Previous       0.96      0.87      0.91        30
       Start       0.74      0.83      0.78        30
        Stop       0.79      0.90      0.84        30
       Hello       0.96      0.83      0.89        30
         Web       0.89      0.57      0.69        30

    accuracy                           0.85       300
   macro avg       0.86      0.85      0.85       300
weighted avg       0.86      0.85      0.85       300
```

Figure 4.29: **Classification matrix of the Kannada dataset with hybrid model**

The classification matrix of Kannada dataset is shown below in the Figure 4.30



```
              precision    recall  f1-score   support

       Avanu       0.78      0.47      0.58        15
       Bagge       0.71      0.67      0.69        15
        Bari       0.48      0.73      0.58        15
     Guruthu       0.50      0.53      0.52        15
        Hogu       0.54      0.47      0.50        15

    accuracy                           0.57        75
   macro avg       0.60      0.57      0.57        75
weighted avg       0.60      0.57      0.57        75
```

Figure 4.30: **Classification matrix of the Kannada dataset with hybrid model**

Method	Accuracy	Dataset
HMM[19]	65%	MIRACL VC1
CNN+GRU[10]	90%	MIRACL VC1
CNN 3D	78.67%	MIRACL VC1
LSTM	66.00%	MIRACL VC1
Hybrid Model	85.00%	MIRACL VC1
CNN 3D	57.33%	Kannada Dataset
LSTM	49.33%	Kannada Dataset
Hybrid Model	61.33%	Kannada Dataset

Table 4.2: **Results of proposed method is compared with existing method for audio visual speech Recognition**

Chapter 5

Conlusion and Future Scope

This chapter concludes the report by giving a conclusion to our project and briefly explains the advantages and disadvantages of the project. Also added the future work possible to improve the model is discussed in this chapter.

5.1 Conclusion

A Visual Speech Recognition (VSR) to interpret the video is being built. Almost all the objectives that were formulated earlier were approached systematically and completed to the fullest. Video recognition for the custom dataset was performed individually for CNN 3D model and LSTM model. Then these two models were combined to a hybrid model which increased the training accuracy significantly. We used the English dataset and achieved the testing accuracy of 79% for CNN 3D model, 66% for LSTM model and 85% for hybrid model(CNN + LSTM). In conclusion, it can be seen that the proposed methodology does outperform other existing methodologies and hence a hybrid model is always the best possible way to achieve high performance measures in Visual Speech Recognition.

5.2 Future Work

There is a huge amount of research in order to improve this model. Some of the future work that can be implemented to improve the proposed model are

- To make the video in different angles other than straight to the face to the speaker.
- To make it able to recognize sentences.

- To make it a real-time model.

5.3 Advantages and Limitations

The following are some of the advantages of VSR model:

- Visual speech recognition provides machines with the ability to understand languages in noisy environments and can also be used for applications related to improved hearing aids and biometric authentication.
- Every module library used in the VSR model, in helping to recognize the word spoken by the speaker, is open source. It is available to everyone.

The following are some of the limitations of VSR model:

- Since accuracy is not exactly 100%, there might be some mistakes while interpreting the word using Visual Speech Recognition.
- If the Audio is also used along with the video, then the accuracy will be better and the system will be more compatible when compared to Visual Speech Recognition.

Reference

[1] . Tatulli and T. Hueber,(2017) "Feature extraction using multimodal convolutional neural networks for visual speech recognition," IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 2971-2975, doi: 10.1109/I-CASSP.2017.7952701

[2] . -H. Goh, K. -X. Lau and Y. -K. Lee,(2019),"Audio-Visual Speech Recognition System Using Recurrent Neural Network," 4th International Conference on Information Technology (InCIT), pp. 38-43, doi: 10.1109/INCIT.2019.8912049.

[3] adayon, Manie Pottie, Greg.(2020), “Comparative Analysis of the Hidden Markov Model and LSTM: A Simulative Approach”, https://arxiv.
org/abs/2008.03825. https://doi.org/10.48550/arXiv.2008.03825.

[4] hillingford, Brendan, Yannis Assael, Matthew W. Hoffman, Thomas Paine, Cían Hughes, Utsav Prabhu, Hank Liao et al.(2018) ,"Large-scale visual speech recognition." arXiv preprint arXiv:1807.05162.https://doi.
org/10.48550/arXiv.1807.05162

[5] . Feng, N. Guan, Y. Li, X. Zhang and Z. Luo,(2017), "Audio visual speech recognition with multimodal recurrent neural networks," 2017 International Joint Conference on Neural Networks (IJCNN), pp. 681-688, doi: 10.1109/IJCNN.2017.7965918.

[6] . Zhou, W. Yang, W. Chen, Y. Wang and J. Jia,(2019), "Modality Attention for End-to-end Audio-visual Speech Recognition," ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 6565- 6569, doi: 10.1109/ICASSP.2019.8683733.

[7] . Petridis, Z. Li and M. Pantic,(2017), "End-to-end visual speech recognition with

LSTMS," 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 2592-2596, doi: 10.1109/ICASSP.2017.7952625.

[8] immermann M., Mehdipour Ghazi M., Ekenel H.K., Thiran JP. (2017), Visual Speech Recognition Using PCA Networks and LSTMs in a Tandem GMM-HMM System. In: Chen CS., Lu J., Ma KK. (eds) Computer Vision – ACCV 2016 Workshops. ACCV. Lecture Notes in Computer Science, vol 10117. Springer, Cham. https://doi.org/10.1007/978-3-319-54427-4_2.

[9] . Tao and C. Busso, (2021),"End-to-End Audiovisual Speech Recognition System With Multitask Learning," in IEEE Transactions on Multimedia, vol. 23, pp. 1-11, doi: 10.1109/TMM.2020.2975922.

[10] . K. Mudaliar, K. Hegde, A. Ramesh and V. Patil,(2020), "Visual Speech Recognition: A Deep Learning Approach," 5th International Conference on Communication and Electronics Systems (ICCES), pp. 1218-1221, doi: 10.1109/ICCES48766.2020.9137926.

[11] handa A., Venkatesan S.M., (2017), Audio Visual Speech Recognition Using Deep Recurrent Neural Networks. In: Schwenker F., Scherer S. (eds) Multimodal Pattern Recognition of Social Signals in Human-Computer-Interaction. MPRSS 2016. Lecture Notes in Computer Science, vol 10183. Springer, Cham. https://doi.org/10.1007/978-3-319-59259-6_9

[12] . Makino et al., (2019), "Recurrent Neural Network Transducer for Audio-Visual Speech Recognition," IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), pp. 905-912, doi: 10.1109/ASRU46091.2019.9004036.

[13] . Petridis, Z. Li and M. Pantic,(2017), "End-to-end visual speech recognition with LSTMS," 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 2592-2596, doi: 10.1109/ICASSP.2017.7952625.

[14] . -H. Goh, K. -X. Lau and Y. -K. Lee,(2019), "Audio-Visual Speech Recognition System Using Recurrent Neural Network," 4th International Conference on Information Technology (InCIT), pp. 38-43, doi: 10.1109/INCIT.2019.8912049.

[15] . Hori, J. Cho and S. Watanabe,(2018), "End-to-end Speech Recognition With Word-Based Rnn Language Models,"IEEE Spoken Language Technology Workshop (SLT), 2018, pp. 389-396, doi: 10.1109/SLT.2018.8639693.

[16] . Amberkar, P. Awasarmol, G. Deshmukh and P. Dave,(2018) "Speech Recognition using Recurrent Neural Networks," International Conference on Current Trends towards Converging Technologies (ICCTCT), pp. 1-4, doi: 10.1109/ICCTCT.2018.85-51185.

[17] . Su, L. Wang and X. Liu,(2017), "Multimodal learning using 3D audio-visual data for audio-visual speech recognition,", International Conference on Asian Language Processing (IALP), pp. 40-43, doi: 10.1109/IALP.2017.8300541.

[18] aind, Sonali B., and Priyanka Wankar.(2014), "Research paper on basics of artificial neural network." International Journal on Recent and Innovation Trends in Computing and Communication 2, no. 1: 96-100.https://doi.org/10.1017/s10499-019-0069-10

[19] oda, K., Yamaguchi, Y., Nakadai, K. et al.,(2015), Audio-visual speech recognition using deep learning. Appl Intell 42, 722–737. https://doi.org/10.1007/s10489-014-0629-7

[20] . Tamura et al.,(2015), "Audio-visual speech recognition using deep bottleneck features and high-performance lipreading," 2015 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA), pp. 575-582, doi: 10.1109/APSIPA.2015.7415335.

[21] akashima, Yuki, Ryo Aihara, Tetsuya Takiguchi, Yasuo Ariki, Nobuyuki Mitani, Kiyohiro Omori, and Kaoru Nakazono.(2016), "Audio-Visual Speech Recognition Using Bimodal-Trained Bottleneck Features for a Person with Severe Hearing Loss." In Interspeech, pp. 277-281.

[22] . M. Fayek, M. Lech and L. Cavedon,(2015), "Towards real-time Speech Emotion Recognition using deep neural networks,"9th International Conference on Signal Processing and Communication Systems (ICSPCS), pp. 1-5, doi: 10.1109/ICSPCS.2015-.7391796.

[23] . Afouras, J. S. Chung, A. Senior, O. Vinyals and A. Zisserman,(2018), "Deep Audio-visual Speech Recognition," in IEEE Transactions on Pattern Analysis and Machine Intelligence, doi: 10.1109/TPAMI.2018.2889052.

[24] handa A., Venkatesan S.M. ,(2017), Audio Visual Speech Recognition Using Deep Recurrent Neural Networks. In: Schwenker F., Scherer S. (eds) Multimodal Pattern Recognition of Social Signals in Human-Computer-Interaction. MPRSS 2016. Lecture Notes in Computer Science, vol 10183. Springer, Cham. https://doi.org/10.1007/978-3-319-59259-6_9

[25] oda, K., Yamaguchi, Y., Nakadai, K. et al.,(2015), Audio-visual speech recognition using deep learning. Appl Intell 42, 722–737. https://doi.org/10.1007/s10489-014-0629-7

[26] . Afouras, J. S. Chung, A. Senior, O. Vinyals and A. Zisserman,(2018), "Deep Audio-visual Speech Recognition," in IEEE Transactions on Pattern Analysis and Machine Intelligence, doi: 10.1109/TPAMI.2018.2889052.

[27] raves, Alex, and Navdeep Jaitly.,(2014), "Towards end-to-end speech recognition with recurrent neural networks." In International conference on machine learning, pp. 1764-1772. PMLR, 2014.

[28] . Afouras, J. S. Chung, A. Senior, O. Vinyals and A. Zisserman,(2018), "Deep Audio-visual Speech Recognition," in IEEE Transactions on Pattern Analysis and Machine Intelligence, doi: 10.1109/TPAMI.2018.2889052.

[29] aeem Ahmed, Mrs. Champa H, (2015), Applying Hidden Markov Model Technique in CSMMI for Action and Gesture Recognition, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH TECHNOLOGY (IJERT) ICESMART (Volume 3 – Issue 19),doi: 10.1109/ICSPCS.2015.7391796.

[30] riyanka T B, Sindhu, Uma D, Varshitha S M, Manjula G, (2021), Speech Enhancement using Hidden Markov Models and Mel-Frequency, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH TECHNOLOGY (IJERT) NCCDS – 2021 (Volume 09 – Issue 12),https://doi.org/10.1007/s10489-014-0629-7