PES University

(Established under Karnataka Act No. 16 of 2013) 100-ft Ring Road,
Bengaluru – 560 085, Karnataka, India

Report on

Visual Speech Recognition

Submitted by

Jayanth S (01FB16EEC111)

Supervised by

Dr. Niranjana Krupa
Professor
Electronics and Communication
PES University

Faculty of Engineering
Electronics and Communication
Bachelor of Technology

Jan - May 2020

CERTIFICATE

This is to certify that the report entitled

Visual Speech Recognition

is a bonafide work carried out by

Jayanth S (01FB16EEC111)

in partial fulfillment for the completion of 8th semester course work in the Program of Study B.Tech in Electronics and Communication Engineering under rules and regulations of PES University, Bengaluru during the period Jan – May 2020. It is certified that all corrections/suggestions indicated for internal assessment have been incorporated in the report. The report has been approved as it satisfies the 8th semester academic requirements in respect of project work.

Signature with date & seal	*Signature with date & seal*	*Signature with date & seal*
Internal Guide	*Chairperson*	*Dean of Faculty*

Name of the student **Jayanth S**

Name of the examiners:

1.
2.
3.

DECLARATION

I, Jayanth S, hereby declare that the report titled '***Visual Speech Recognition***', is an original work done by me under the guidance of **Dr. Niranjana Krupa**, Professor of Electronics and Communication, PES University, and is being submitted in partial fulfillment of the requirements for completion of 8th Semester course work in the Program of Study B.Tech in Electronics and Communication Engineering.

PLACE: BENGALURU

DATE:

NAME AND SIGNATURE OF THE CANDIDATE

JAYANTH S

Abstract

Visual speech recognition is the task of utilizing the specific facial configuration made while speaking to predict the word being uttered. It has become an increasingly more popular task, especially after the advent of deep learning models, as a way to augment automatic speech recognition, which is becoming saturated in terms of performance. There are broadly two approaches to VSR, the visemic approach and the holistic approach, each with its own pros and cons. This work primarily deals with the visemic approach in which visemes are predicted, which can later be combined to predict words. It describes, implements and trains several deep neural networks for VSR. Firstly, bare Convolutional Neural Networks (CNN) are trained to establish their baseline accuracies. Secondly, more sophisticated models, such as the Long Short-Term Memory (LSTM) and the Hidden Markov Model (HMM), are built to try and increase these base performances. Additionally, an innovative approach is tried and tested in which generative adversarial networks(Generative Adversarial Networks (GAN)s) are used to augment the dataset by creating new images instead of the traditional augmentation methods. All these methods are trained and evaluated on the TCD-Timit corpus of continuous speech, which was preprocessed to extract images. Finally, a comparison of all the models is made. The results demonstrate that the CNN-LSTM hybrid model performs better than the other models, achieving an average accuracy of 55.27% on two speakers. Also, the approach that involved augmentation using images generated by the GAN shows promise as it led to a considerable increase in performance over the baseline models.

Acknowledgement

This project was part of the 8th semester BTech(ECE) curriculum. Implementing it over the past 5 months has been an enriching experience as it was a great chance for learning and personal development.

I would like to express my sincere gratitude to my guide Professor Niranjana Krupa for her constant guidance and overwhelming support throughout the project. I would also like to thank my panelist, Professor Preethi S J, for her expert inputs.

Further, I would like to extend my thanks to Professor Anuradha M, Chairperson, and Staff of the Electronics and Communication Department for helping me and providing the tools required to execute the project.

I would like to express my sincere gratitude to the Principal Dr. Sridhar, Pro-Chancellor Prof. D Jawahar, Vice-Chancellor Dr. J Suryaprasad, and to Chancellor of PES Institutions, Dr. M R Doreswamy for providing the opportunity to be a part of this prestigious Institution and exposing me to various educational domains.

Finally, I would like to thank my Parents for their unwavering support over the course of this project, without whose backing this would not have been possible.

Contents

List of Abbreviations

VSR	Visual Speech Recognition
ASR	Automatic Speech Recognition
HMM	Hidden Markov Model
RNN	Recurrent Neural Networks
LSTM	Long Short-Term Memory
GAN	Generative Adversarial Networks
CNN	Convolutional Neural Networks
GRU	Gated Recurrent Units
VGG	Visual Geometry Group
BBC	British Broadcasting Corporation
LRW	Lip Reading in the Wild
ST-CNN	Spatiotemporal Convolutional Neural Networks
DCGAN	Deep Convolutional Generative Adversarial Networks
CT	Computed Tomography
ILSVRC	ImageNet Large Scale Visual Recognition Challenge
RBM	Restricted Boltzmann Machines
SMOTE	Synthetic Minority Over-sampling Technique
ReLU	Rectified Linear Unit
DWT	Discrete Wavelet Transform
DCT	Discrete Cosine Transform
PCA	Principle Component Analysis
AVI	Audio Video Interleave
PGGAN	Progressively Growing Generative Adversarial Network

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Speech recognition systems [4],[5],[6] are pivotal for effective human-computer interaction and have been a popular topic of research for the past three decades. However, vanilla speech recognition has two fundamental shortcomings. Firstly, acoustic variations, due to both the environment and the characteristics of the vocal tract, seriously degrade speech recognition systems. This problem has been explored rigorously and various types of noise-robust speech recognition systems [7],[8],[9] have been proposed. Secondly, it does not utilize visual information, unlike humans. As humans interpret speech using not only audible information but also the visual cues associated with it, incorporating vision into human-machine interfaces is a natural advancement in this technology. However, most current systems work on word-wise speech recognition that needs a language model and an extensive dictionary. Therefore, these systems are language dependent. Speech recognition at a more fundamental level, that is, the phoneme-viseme level will forego the need for systems that are so heavily language dependent, as all languages have pretty much the same set of phonemes, give or take a few. This project focuses on the phonemic/visemic approach for Visual Speech Recognition (VSR) and aims to build efficient phoneme and viseme recognition models to aid in VSR.

1.2 Problem Statement

Visual speech recognition [10],[11], popularly called lip reading, is the process of understanding speech uttered by an individual by interpreting the movement of the mouth region. It is widely used by hard-of-hearing people to better perceive speech, and involuntarily by everyone in environments where audio speech recognition is hampered by noise. Phonemes are the smallest distinct parts of speech that help distinguish between words of a language. A viseme is a generic snapshot of the face that is made when a phoneme is uttered. Visual speech recognition deals with utilizing visemes to aid in speech recognition and therefore, it fundamentally incorporates auditory noise-robustness to a certain extent and can be a solution to the former problem in automatic speech recognition as mentioned above, while simultaneously providing more information to the machine for achieving better recognition accuracy. It is essentially a computer vision task as the machine utilizes image frames from a video to capture facial signals and emotions, which will provide it with additional information, thereby increasing recognition accuracy of the phoneme being uttered. These systems, called audio-visual speech recognition systems [12],[13], have already been implemented and tested over a wide range of tasks.

However, Visual Speech Recognition (VSR) has not become as popular as Automatic Speech Recognition (ASR) and researchers have often claimed that the unavailability of a large, suitable database for training is the reason for VSR to lag behind ASR. Also, the lack of a standard database has been cited as a major bottleneck for the comparison of these audio-visual recognition algorithms by Potamianos et al [14]. Furthermore, a major deficit of any database for VSR is that the visemes are unequally represented as expected of speech patterns found in the wild. As a consequence, it becomes inherently harder to train a convolutional network to recognize the lesser-used visemes.

1.3 Main Objectives

The main objectives of this project are twofold. Firstly, it aims to describe and implement several deep learning network architectures that are currently in use in the field of lip reading, namely CNN [15], Recurrent Neural Networks (RNN) [16], particularly LSTM [17], and Hidden Markov Models (HMMs) [18]. Additionally, a comparison of the performances of these models is presented.

The second objective of this project is to demonstrate the use of Generative Adversarial Networks (GAN) [19] as a tool for data augmentation which entails the design, implementation, and training of a GAN that is capable of generating images. This network will then be used to synthesize new images and thus

increase the size of the dataset, as well as lessen the imbalance in size among all the viseme classes in the hope that it will improve classification performance.

1.4 Methodology

The methodology followed in this project is as follows: Pre-processing was done on the TCD-Timit database to obtain input images of the desired dimensions. Then, the baseline accuracies of VGG16 and AlexNet on both phonemes and visemes was established. Finally, three different architectures that might boost performance of the CNNs were implemented:

1. A CNN-LSTM hybrid model using AlexNet
2. A CNN-HMM hybrid model using VGG16
3. Utilizing GANs as a tool for data augmentation and increasing the size of the dataset

A block diagram capturing the complete flow of the project is presented in Figure 1.1.

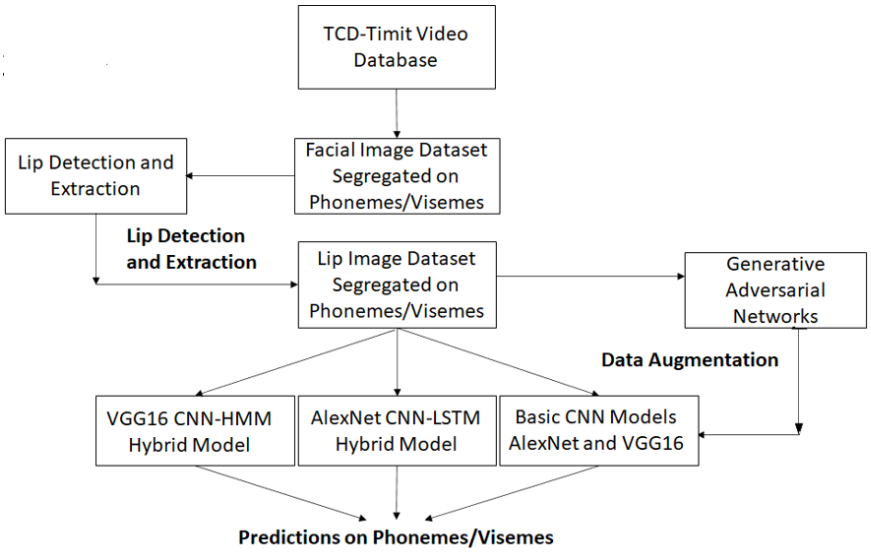

Figure 1.1: The Complete Flow of the Project

1.5 Scope of the Project

A machine that is capable of comprehending lip movement has many great applications. It can be used in biometric face authentication to reinforce security measures. It is a powerful tool in multi-speaker speech decoding, that is, to understand two speakers who are speaking simultaneously and forensics. It will be also be useful in increasing the capabilities of speech recognition systems in general. Furthermore, lip reading systems can be used as an external device by others to interpret the speech of people suffering from damaged vocal tracts. This is a necessity as lip reading is a notoriously difficult task as an average, hearing-impaired lip reader is capable of correctly interpreting small sentences (of 6 words) only 52.3% of the time [20].

1.6 Organization of Thesis

Having discussed the problem of visual speech recognition and presented an overview of the objectives, the process of design, implementation and training of the networks used will be described in detail in the coming chapters.

In Chapter 2, an overview of some of the research conducted in lip reading until now is presented in the first half. The use of convolutional neural networks is first described, after which deeper networks involving LSTMs, autoencoders, Gated Recurrent Units (GRU) and HMMs are discussed. The second half focuses on the field of data augmentation and the various methods used, along with a review of the use of GANs in this field. The chapter ends with a specific example of the use of GANs as a tool for data augmentation in the field of VSR.

Chapter 3 describes the original video dataset and then details the preprocessing steps that were taken to create an image dataset consisting of images of the lip region of each speaker. In Chapter 4, a detailed summary of all the network architectures that are used in this project is presented. Chapter 5 is used to describe the training of these networks and the network parameters used for each of them.

Chapter 6 compares the performances of the networks and provides insights as to why some networks perform worse than anticipated. Chapter 7 concludes the project and outlines further work that can be undertaken to improve classification performance in visual speech recognition.

Chapter 2

Literature Survey

2.1 Visual Speech Recognition

Humans, with the help of vision, learn to understand speech uttered by a person which is an audio signal, that is, they perform audio-visual speech recognition. For the hearing impaired, the visual aspect of the speaker plays a dominant part in speech recognition and this action, in layman terms, is known as 'lip reading'. In [21], simple stochastic networks such as an HMM is used to classify a digit uttered by a speaker. The dataset contained 12 speakers(9 male and 3 female) speaking the digits 1, 2, 3 and 4 twice. The accuracy obtained was comparable to that achieved by an average untrained human being.

Lip reading done using machine learning or deep learning is known visual speech recognition.There are basically two approaches to VSR. They are the visemic and holistic approaches.In the visemic approach, the VSR system tries to recognize the viseme being uttered by a speaker. In the holistic approach, the VSR system is trained to recognize complete words/sentences uttered by the speaker. Detailed explanation on both the approaches with their strengths and weaknesses can be found in [22].

2.2 CNNs and RNNs used in VSR

Visual speech recognition is not a new problem and has been an area of interest in computer vision for a few decades. Yehia et al. [23] established the relation between visual and auditory features where they found a strong correlation

between the two. Thereafter, various researchers have modeled this audio-visual relationship. Early models, such as the ones employed by Matthews et al. [24] and Zhao et al. [25] did not employ deep learning. Their works involve extensive preprocessing for detecting the mouth region and then extracting its features. In recent years, deep learning techniques have been applied to the problem of visual speech recognition. Garg et al [26] designated it as a classification problem and proposed two different techniques for word-level lip reading on the MIRACL – VC1 [27] dataset. The first method involved concatenating all frames that made up a word sequence into a single larger image and feeding it to a pre-trained Visual Geometry Group (VGG) network [28] convolutional neural network pretrained on faces while the second method entailed extracting features using the VGGNet and passing these through LSTM [17] layers for classification. They achieved validation accuracies of around 76% and 30% respectively for the two methods.

Lee et al. in [29] utilize multiple views of each speaker in the OuluVS2 [30] public database to develop a phrase-level multi-view lip reading model using end-to-end neural networks consisting of custom-made convolutional neural networks and LSTMs [17], the intuition being that different views of the same utterance provides more information to the neural network model, thus boosting its performance. They present three different lip reading tasks : the single-view task in which the model is trained and tested on a single view, the cross-view task in which the model is trained on all views together and then tested separately, and finally the multi-view task in which the model is trained and tested on synchronized data that was recorded from multiple views. They recorded average accuracies of 77.9%, 83.8% and 78.6%, proving definitively that multiple views provide more features to the network which leads to increased performance.

Convolutional auto encoders are used by Parekh et al. [31] for extracting features that are then fed to an LSTM network. They preprocess the input to contain only the lip region and train the convolutional autoencoder on these images. Once trained, they discard the decoder and use the encoder to extract features in sequence. These features are fed into an LSTM classifier. Their model achieved a maximum word-level test accuracy of 85.61% on nine words chosen from the British Broadcasting Corporation (BBC)'s Lip Reading in the Wild (LRW) dataset [32]. Further, they achieved test accuracies of 63.22% and 84.8% on small datasets derived from the MIRACL-VC1 and GRID databases [33] respectively.

Similar to the auto encoders used by Parekh et al. [31], Petridis [34] used an encoder consisting of 3 sigmoid layers pre-trained in a greedy layer-wise manner using Restricted Boltzmann Machines (RBM).The architecture had two streams, one of which is trained on preprocessed lip images and the other on differences between lip images of two consecutive frames. These features are fed into their respective LSTM classifiers, which are later concatenated and then fed into a Bi-LSTM layer. Their network achieved a maximum phrase level test accuracy

of 84.5% on the OuluVS2 [30] database and a digit classification test accuracy of 78.6% on the CUAVE [35] database.

Assael et al. proposed LipNet [20], the first end-to-end sentence-level lip reading model in 2016 that outperformed all other models and even human lipreaders by achieving a sentence-level accuracy of 95.2% on the GRID [33] corpus. They implemented a model made up of Spatiotemporal Convolutional Neural Networks (ST-CNN) [36], two bi-GRUs [37] and the connectionist temporal classification loss. The features extracted using the ST-CNN are fed into the bi-GRUs for temporal aggregation of the data. Furthermore, their model is capable of handling variable length sequences. Also, their model exhibited a low word error rate of 11.4% and an even lower character error rate of 6.4% for unseen speakers.

2.3 Hidden Markov Models used in VSR

All of the above works performed either word, phrase or sentence level classification. In contrast, Noda et al. [38] in 2014 proposed the use of CNNs for visual phoneme recognition. Their dataset consisted of 1.23×10^5, 32×32 images of the mouth region belonging to 6 different speakers. They trained 6 separate AlexNet [39] convolutional neural networks for each speaker to recognize 40 different phonemes, and achieved an average phoneme recognition accuracy of 58%. In 2017, Tatulli et al.[40] presented a CNN - based feature extractor and a multistream GMM-HMM model for phoneme-wise classification. Their dataset consisted of images of the mouth region along with the corresponding ultrasound images of the tongue. They performed their experiments on four different architectures each of which incorporated a CNN model and a GMM-HMM model. Best performance was obtained using a multi-modal CNN architecture using a middle fusion strategy where they achieved an accuracy of 80.4%.

In 2014, Wu et. al in [41] presented a lip reading technique that used cascade feature extraction and HMM. They constructed a Chinese Audio-Video database which included 37 frequently-used Chinese characters for recognition. All the videos were recorded at 25 frames per second with a resolution of 720×576. Speaker dependent experiments were conducted on each person with all the test samples. They used 4 different methods for feature extraction, namely, Discrete Cosine Transform (DCT), Discrete Wavelet Transform (DWT), DCT-Principle Component Analysis (PCA) and DWT-PCA. They created feature vectors for each speaker using each of these methods and then fed this as input to the HMM. DWT-PCA showed the best performance among the four, achieving an accuracy of 77.4%.

The aim of [42], presented by Sujatha et. al, was the extraction of visual

lip movements (lip features) and prediction of the word which was pronounced. The participants were 4 females and 6 males, distributed over different age groups. The videos were recorded at 25 frames per second, stored in Audio Video Interleave (AVI) format and resized to 320×240 pixels. Each speaker in each recorded video uttered 35 different words 20 times non-contiguously. They created a HMM system consisting of 35 HMM models to recognize each of the 35 words and utilized the 16 point DCT for feature extraction. They achieved a word recognition accuracy of 97.5%.

2.4 GANs as a tool for Data Augmentation

Affine transformations are the most commonly used methods of data augmentation and are used to synthetically increase the size of the dataset by performing basic image manipulations such as translation, scaling, rotation, shearing etc. on the images in the original dataset. These techniques have been successful in preventing overfitting and thus increasing classification accuracies in convolutional neural networks where techniques like dropout[43] and batch normalization[44] cannot be used due to limited dataset sizes. A naive way to balance the dataset across classes was to simply replicate the images in the minority dataset. A more intuitive way of balancing the dataset was proposed by Chawla et al.[45] in 2002. Their technique, called Synthetic Minority Over-sampling Technique (SMOTE), proposed a combination of over-sampling the minority classes while simultaneously under-sampling the majority classes to improve classification accuracy.

The advent of generative adversarial networks has enabled new methods of data augmentation through generative modelling. They have become popular in the field of medical imaging where datasets are small in size. Wu et al.[46] proposed the use of a conditional infilling GAN to produce high-resolution synthetic mammogram patches. Frid – Adar et al. [47] used Deep Convolutional Generative Adversarial Networks (DCGAN) [48] to generate synthetic images of liver lesions. Their dataset was very limited and consisted of 182 computed tomography images. They recorded an improved sensitivity of 85.7% and specificity of 92.4% on the dataset augmented using GANs as opposed to a sensitivity of 78.6% and a specificity of 88.4% on the dataset augmented using classic data augmentation techniques. Chuquicusma et al.[49] proposed the use of DCGANs to generate images of lung nodules. Their dataset consisted of 1018 lung cancer screening thoracic Computed Tomography (CT) scans. Two radiologists took part in 18 visual Turing tests to check whether the generated images were of high quality and were good enough to pass as real samples. They recorded that their generated images managed to fool radiologist 1 in 67% of their experiments while they managed to fool radiologist 2 in all of their experiments.

In the field of lip reading, Oliviera et al. [50] utilize the Pix2Pix GAN to

map between views of the mouth taken from random angles to the frontal view. Their generator receives an image taken from a random angle and generates the corresponding image as seen from the frontal view. The discriminator is trained on the pairs of images generated by the generator (that consist of a real image and its corresponding generated frontal view) and the original pairs, so that it learns to classify between them. They then performed data augmentation on the original dataset by concatenating real and generated views and fed it to Xception architecture for classification. This method of utilizing multiple views for viseme classification resulted in an average increase in accuracy of 5.9% on the GRID corpus.

Chapter 3

Dataset Description and Pre-processing

3.1 The Dataset

The TCD-Timit audio-visual corpus of continuous speech [51] is a publicly available video database of 62 speakers uttering a total of 6913 phonetically diverse sentences, of which 3 are professionally trained lip speakers. Videos were recorded at a resolution of 1920 × 1080-pixel frames at 30 fps from two different angles – frontal (0 degrees) and 30 degrees. All experiments done in this paper were conducted on images extracted from the videos of 2 different speakers, a male and a female, both of whom were not lipspeakers. A sample frame taken from a video of the male speaker "Speaker 1" is shown in Figure 3.1.

Figure 3.1: A frame of the male speaker 01M taken from a video in TCD-Timit

3.2 Dataset Preprocessing

Preprocessing was done on 98 videos each taken from the frontal view of 2 speakers. All frames from each video of a total of 15 speakers were extracted and stored in the corresponding phoneme folder with the help of the phoneme transcription label files provided with the TCD-Timit database. There were a total of 38 phonemes including the silence /sil/ phoneme. However, the /sil/ phoneme was not included in the experiments. as it would lead to further ambiguity among the /m/, /b/, /p/ and /sil/ phonemes. The OpenCV [52] library was used to code the lip detection and extraction algorithms such as the one in [53], which were used to extract only the lip region of each image to obtain a new image dataset consisting of the lip region for each speaker. All lip images were further resized to 256×256 dimensions. Figure 3.2a depicts the cropped image of Figure 3.1 and Figure 3.2b depicts a sample, final resized image the likes of which makes up the lip dataset.

(a)

(b)

Figure 3.2: Pre-processed Images: (a) Lip Image cropped from Figure 3.1; (b) Lip Image in 3.2a resized to 256×256 dimensions

Neti's phoneme-viseme mapping [12] as shown in Figure 3.3 was used to merge phoneme folders with similar visemes (such as /b/ and /p/) into a smaller set of 12 viseme class folders. For feeding the lip images to the CNN – LSTM network, videos were reconstructed using the lip images only.

Silence	/sil/, /sp/
Lip-rounding based vowels	/ao/, /ah/, /aa/, /er/, /oy/, /aw/, /hh/ /uw/, /uh/, /ow/ /ae/, /eh/, /ey/, /ay/ /ih/, /iy/, /ax/
Alveolar-semivowels	/l/, /el/, /r/, /y/
Alveolar-fricatives	/s/, /z/
Alveolar	/t/, /d/, /n/, /en/
Palato-alveolar	/sh/, /zh/, /ch/, /jh/
Bilabial	/p/, /b/, /m/
Dental	/th/, /dh/
Labio-dental	/f/, /v/
Velar	/ng/, /k/, /g/, /w/

Figure 3.3: Neti's Phoneme-Viseme Mapping

Chapter 4

Network Architectures

4.1 VGG16 Convolutional Neural Network

VGG16 is a convolutional neural network model that was proposed by Simonyan and Zisserman in [28]. It is best known for achieving a 92.7% top-5 test accuracy on the ImageNet dataset [54], a dataset consisting of 14 million images belonging to 1000 different classes, at the ImageNet Large Scale Visual Recognition Challenge (ILSVRC)–2014, a significant improvement over AlexNet. VGG16 has different configurations as described in the paper. The architecture of VGG16 used in this project is as shown in Figure 4.1 . Note that the fully connected layers containing the weights pre-trained on ImageNet were removed and a new stack was added to learn a new set of weights pertaining to the lip images dataset. This process is known as transfer learning as the network rapidly learns in the new task through the transfer of knowledge from a related task (here, it is the training of the network on ImageNet).

4.2 AlexNet Convolutional Neural Network

Two deep convolutional networks have been used for classification, one of them being AlexNet [39] which is pre-trained on the ImageNet [54] dataset.It has 4 convolutional layers with 96 filters for the first convolutional layer, 256 filters for the second, and 384 for the 3rd, 4th and 5th layers. Each convolutional layer is followed by a Rectified Linear Unit (ReLU) [55] to avoid saturation of the neurons. This is followed by cross channel normalization for reducing the number of epochs required for training. Maxpooling is introduced after the 1st,

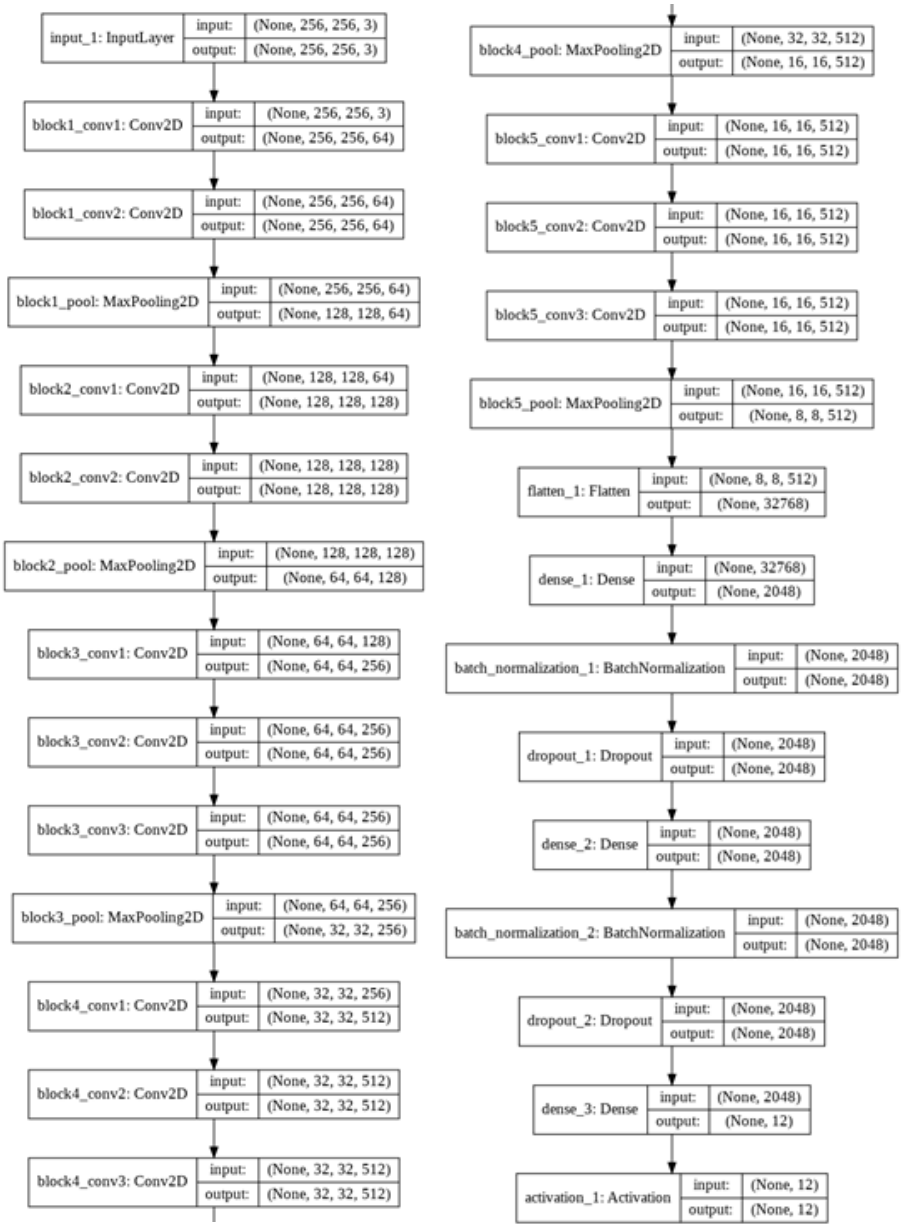

Figure 4.1: VGG16 Architecture used in this project

2nd and 5th convolutional layers to negate the effect of shifting and rotation of images used for training. Dropout [43] is present after the respective the convolutional segment to avoid overfitting. The above mentioned layers are used for feature extraction and classification, while two fully connected layers with the softmax activation function are used along with dropout to avoid overfitting. The architecture of AlexNet taken from [56] is shown in Figure 4.2.

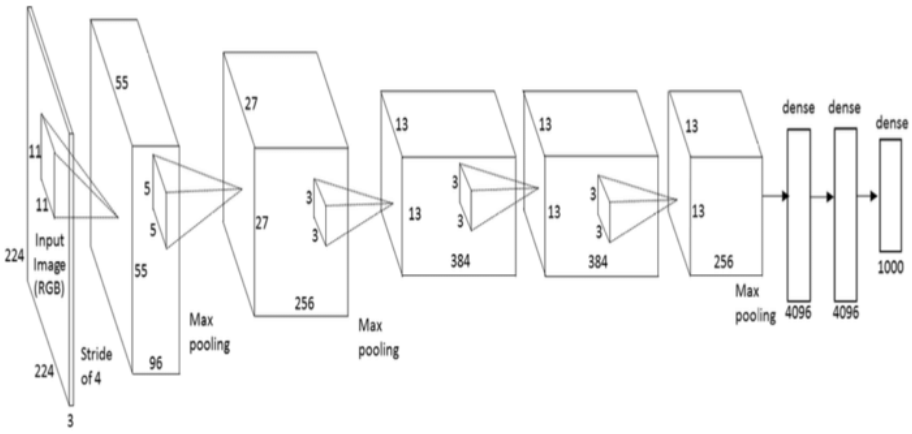

Figure 4.2: AlexNet Architecture

4.3 Recurrent Networks and Long Short-Term Memory Networks

Phoneme utterences occupy more than a single frame in a video. Furthermore, the number of frames occupied by two phoneme utterences of the same kind can be variable. To account for this temporal aspect of the data, a network capable of learning dynamically over time has to be used. Recurrent neural networks, shown in Figure 4.3, are a class of artificial neural networks having this capability. Additionally, an RNN has an internal memory state that allows it to process variable length sequences.

However, vanilla RNNs exhibit the vanishing gradient problem and the Long Short-Term Memory (LSTM) network was designed to deal with this shortcoming. Each LSTM cell takes into account the previous step output for calculating the present output and thereby incorporates the temporal aspect of the dataset. Figure 4.4, presents a typical LSTM cell along with the operations used within.

The first part of an LSTM is the forget gate layer which takes X_t(the current

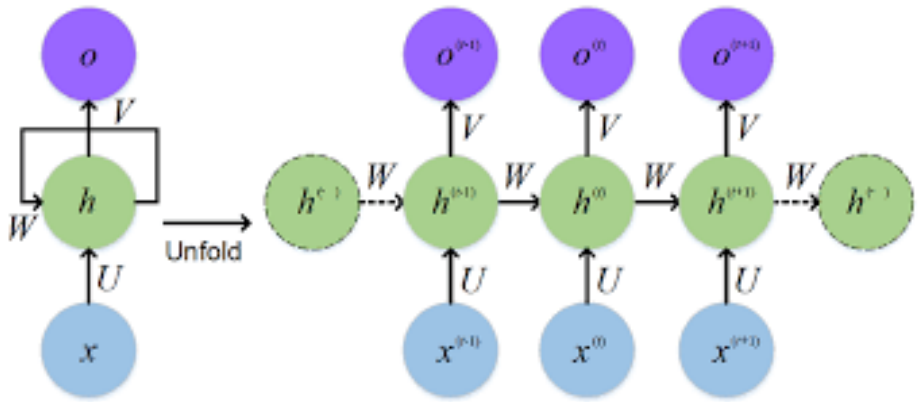

Figure 4.3: A typical RNN. Source:[1]

input) and h_{t-1}(The previous hiddent state) as input and calculates how much of the output as follows:

$$f_t = \sigma(W_f.[h_{t-1}, x_t] + b_f) \tag{4.1}$$

where σ is the sigmoid activation function, W_f is the weight and b_f is the bias. The output of this layer is a value between 0 and 1,where 0 signifies forget everything from the previous state and 1 signifies remember everything from the previous state.

The next step is to decide what new information is going to be stored in the cell state. First, a sigmoid layer called the input gate layer decides which values will be updated. Next, a tanh layer creates a vector of new candidate values $\tilde{C}_t$ that could be added to the state.

$$i_t = \sigma(W_i.[h_{t-1}, x_t] + b_i) \tag{4.2}$$

$$\tilde{C}_t = tanh(W_C.[h_{t-1}, x_t] + b_C) \tag{4.3}$$

In the next step, the cell state C_{t-1} is updated to state C_t using:

$$C_t = (f_t \times C_{t-1}) + (i_t \times \tilde{C}_t) \tag{4.4}$$

Finally, the output that is emitted will be based on the cell state but is a filtered version. A sigmoid layer decides which parts of the cell state are going to be output. Then, the cell state is passed through a tanh layer that condenses the values to between -1 and 1. At the end, this is multiplied with the output

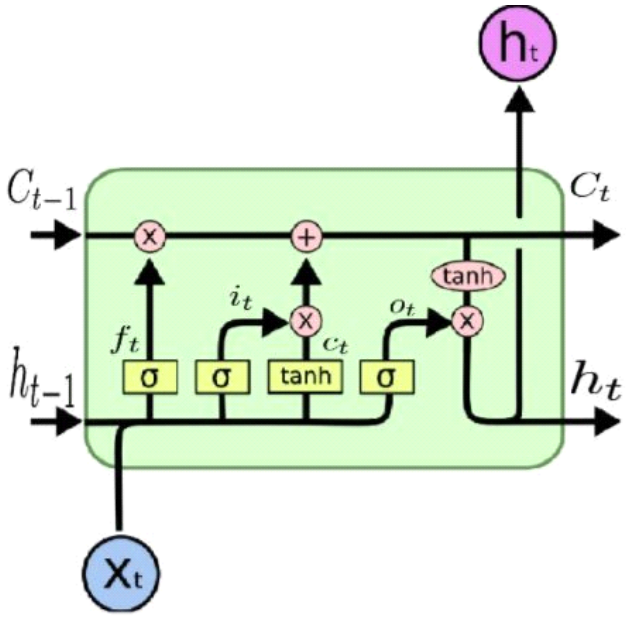

Figure 4.4: A single LSTM cell. Source:[2]

of the sigmoid gate to filter the parts that have to be emitted.

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{4.5}$$

$$h_t = \sigma_t \times tanh(C_t) \tag{4.6}$$

In this project, the inputs to the CNN-LSTM hybrid network were a set of videos that are reconstructed from the lip image dataset. The LSTM network used in this project has two LSTM layers along with dropout and a fully connected layer with the softmax activation function for classification.The number of LSTM neurons and the dropout ratio are varied and the results are discussed in section 5.3. The complete CNN-LSTM network is shown in Figure 4.5.

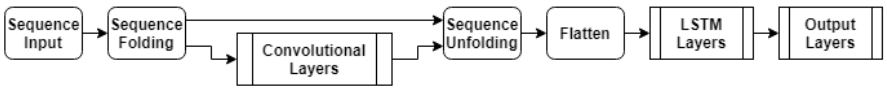

Figure 4.5: The CNN-LSTM Network

4.4 Hidden Markov Models

A Hidden Markov Model (HMM)[57][58][59] is a model that is used to describe the evolution of observable events that depend on hidden factors, which are not observable directly. The hidden factor is called a 'state'. The hidden states form a Markov chain, and the probability distribution of the observed symbol depends on the underlying state.

Due to these properties, HMMs have been extensively used in speech recognition problems. Similarly, an HMM model can also be built for visual speech recognition. This approach is also useful in modelling biological sequences, such as proteins and DNA sequences.

Phoneme utterances occupy more than a single frame in a video, so one image frame cannot decide to which phoneme class it belongs to; a phoneme depends on a series of images in a particular order. To account for this temporal aspect of the data, HMM models can be used.

The different components of a HMM are the hidden states (Q), the observations (O) , the transition probability matrix (A) , the emission probabilities (B) and an initial starting probability distribution (π). With all this in mind, an HMM can be defined as follows:

Table 4.1: Definition of a Hidden Markov Model (HMM)

$Q = q_1 q_2 ... q_N$	the set of N states
$O = o_1 o_2 ... o_T$	a sequence of T observations
$A = a_{11} ... a_{ij} ... a_{NN}$	the transition probability matrix A. Each element a_{ij} denotes the probability of moving from the present state i to the next state j, such that $\sum_{j=1}^{N} a_{ij} = 1 \quad \forall i$
$B = b_i(o_t)$	a sequence of emission probabilities, each denoting the probability of an observation o_i being emitted from a state i
$\pi = \pi_1, \pi_2, ..., \pi_N$	the initial probability distribution for each state. Each element π_i denotes the probability that the Markov chain will start at state i. Note that this can also be 0 and $\sum_{i=1}^{N} \pi_i = 1$

An example of a HMM is shown below in Figure 4.6:

Additionally, an HMM makes two assumptions. Firstly, it incorporates the same rule as a Markov chain - the probability of a particular state depends only on the previous state, that is, the Markov assumption given by:

$$P(q_i|q_1 ... q_{i-1}) = P(q_i|q_{i-1}) \tag{4.7}$$

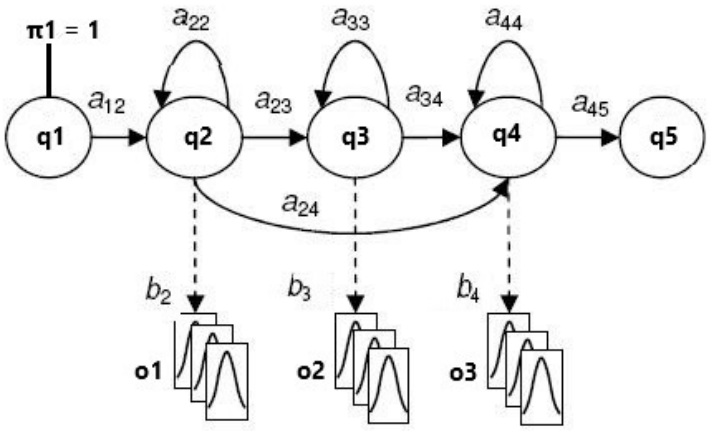

Figure 4.6: A Hidden Markov Model

The second assumption is the output independence assumption that states that the probability of an output observation o_i depends only on the state that produced the observation q_i and not on other states or observations:

$$P(o_i|q_1...q_i, ..., q_T, o_1, ..., o_i, ..., o_T) = P(o_i|q_i) \tag{4.8}$$

There are 3 basic problems when dealing with HMMs. A brief summary along with their algorithms are presented below:

1. Trying to figure out what is the probability with which a given model consisting of $\lambda = (A, B, \pi)$ generates a sequence of observations $O = o_1, o_2, o_3, \ldots, o_n$, which is denoted by $P(O|\lambda)$.
   To solve this problem, the forward algorithm which defines and calculates the essential parameter α, the forward parameter, is used. It is a dynamic programming solution that computes the forward trellis $\alpha_t(j)$. The forward parameter α for each cell in the trellis represents the probability of ending up in that state while following λ, given by

   $$\alpha_t(j) = P(o_1, o_2...o_t, q_t = j|\lambda) \tag{4.9}$$

   where $q_t = j$ means "the t^{th} state in the sequence of states is state j". The computation of the probability value $\alpha_t(j)$ is done by by summing over the extensions of all the paths that lead to the current cell. For a given state q_j at time t, the value $\alpha_t(j)$ is computed as

   $$\alpha_t(j) = \sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij} b_j(o_t) \tag{4.10}$$

2. Finding out what sequence of states best explains the sequence of observations or, more precisely, what is the most likely sequence of states that was traversed to get that particular sequence of observations. This problem is called the decoding task and can be solved by using the popular Viterbi algorithm, another dynamic programming algorithm.

   Each cell in the dynamic programming trellis $v_t(j)$ represents the probability that the HMM is in state j after seeing the first t observations and passing through the most probable state sequence $q_1, ..., q_{t-1}$ only, given the automaton λ. This is shown in the expression:

   $$v_t(j) = \max_{q_1,\ldots,q_{t-1}} P(q_1...q_{t-1}, o_1, o_2...o_t, q_t = j|\lambda) \tag{4.11}$$

   Succinctly put, the Viterbi probability is computed by taking the most probable of the extensions of the paths that lead to the current cell, that is, for a given state q_j at time t, the value $v_t(j)$ is computed as

   $$v_t(j) = \max_{i=1}^{N} v_{t-1}(i) a_{ij} b_j(o_t) \tag{4.12}$$

3. The third and final problem is perhaps the hardest: to learn the parameters A and B of an HMM. The standard approach to this problem is the forward-backward algorithm, popularly called the Baum-Welch algorithm, which is a special case of the expectation-maximization algorithm. Without going into the intricacies of this problem, note that this algorithm is an iterative algorithm, meaning that it computes an initial estimate of the probabilities, and continuously improves on this to get better estimates.

The VGG16 CNN, described in section 4.1, was used to create a hybrid CNN-HMM in this project. The HMM model takes sequences as input. These sequences are generated using the CNN model by extracting features from sequences of image frames of one of the 37 phonemes present in the dataset. The input dataset consists of a set of image frames grouped based on the 37 phonemes and their occurrences in multiple sentences. These image sequences are fed into VGG16 in the exact order in which they appear in the video to preserve temporal information. The CNN outputs features corresponding to these image sequences, which are then fed to the HMM models. VGG16 , therefore, acts as a feature extractor.

Pre-processing was required to feed these feature sequences into the HMM models as only certain data formats are accepted. A total of 37 HMM models, one for each phoneme were implemented as shown in Figure 4.7. The dataset is divided into training and testing data in the ratio 9:1. During testing, features from the sequences were extracted using the CNN and these were fed to every single HMM model. Therefore, each testing datapoint would have 37 probabilities corresponding to the 37 phonemes. Finally, log probability was calculated

for each and the phoneme corresponding to the highest log probability value was declared the final prediction. This was repeated for a set of visemes that were grouped using the Neti map as shown in Figure 3.3. Results for both the phoneme and viseme groupings are discussed in section 5.4.

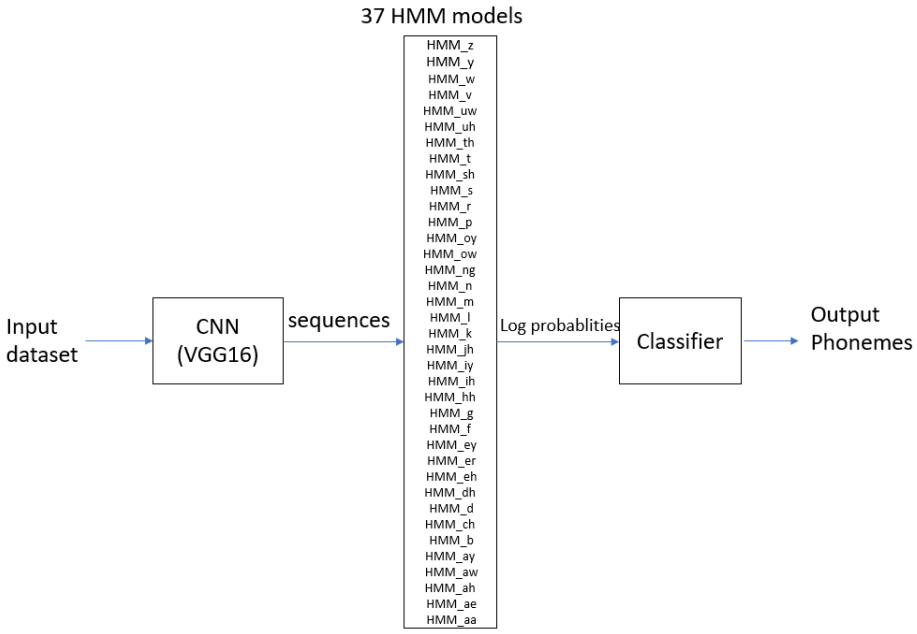

Figure 4.7: Hybrid CNN - HMM Model used in this project

4.5 Generative Adversarial Networks

Generative Adversarial Networks (GAN)[19] are a relatively new type of neural network architecture that have become extremely popular as a tool for data augmentation. They consist of two competing neural networks, the generator and the discriminator, taking part in a game. The generator network G creates images using a latent space of noise. The discriminator network D is fed either a generated image or a real image and attempts to classify whether the image is real (true data sample) or fake (generated by the generator). The gradients of both the networks are then used to propel them in the right direction. Academically, this interaction between the two networks can be described by the minimax objective function:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{\text{data}}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))] \quad (4.13)$$

where $x \sim p_{\text{data}}(x)$ denotes samples x drawn from the real dataset
$z \sim p_z(z)$ denotes samples z drawn from the generator
$D(x)$ represents the output of the discriminator on real data
$D(G(z))$ represents the output of the discriminator on fake data.

After extensive training, the generator becomes capable of producing images which have a distribution indistinguishable from the distribution of the images used for training the discriminator, thus fooling the discriminator successfully. A typical GAN architecture framework is given in Figure 4.8.

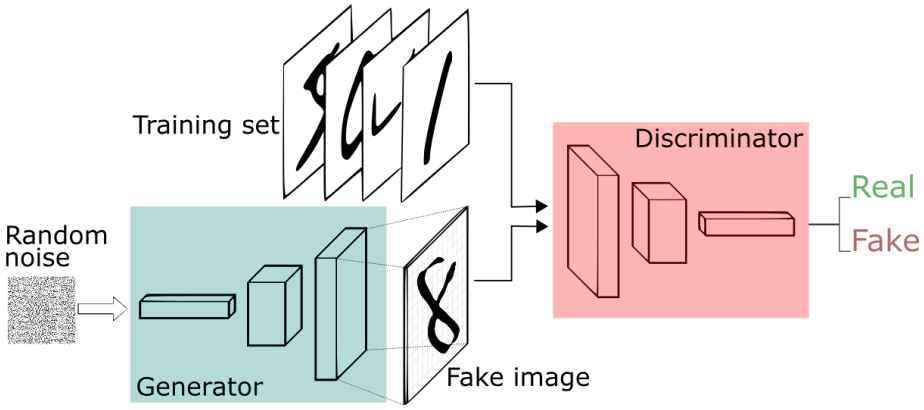

Figure 4.8: Framework of a GAN. Source:[3]

Conditional GANs were an extension of the GAN architecture introduced by Mirza et al. in [60] in which they provide a second input y to both the networks in the form of an additional input layer that is used as conditioning information.

This can later be used to control the data generation process. Theoretically, this can be expressed by the equation:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{\text{data}}(x)}[\log D(x|y)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z|y)))] \tag{4.14}$$

4.6 Deep Convolutional Generative Adversarial Networks

Deep Convolutional Generative Adversarial Networks (DCGAN) [48] were proposed by Radford et al. in an attempt to generate high resolution images using deeper generative models. They modified the original GAN architecture in 3 significant ways:

- Using strided convolutions for spatial downsampling in the discriminator instead of pooling layers and using fractional-strided convolutions in the generator for upsampling.
- Eliminating the use of fully connected layers on top of the convolutional layers
- Utilizing batch normalization [44] for normalizing input and stabilizing training

The DCGAN architecture used in this project was slightly modified as it did not have batch normalization layers [44] and used Leaky Relu [61] in the generator along with gaussian noise layers. Also, the DCGAN was conditioned on the viseme class labels so as to facilitate generation of images belonging to a specific viseme class as required. The architecture of the generator of the conditional DCGAN used to produce images of dimensions $8 \times 8 \times 3$ is depicted in Figure 4.9 and the architecture of the corresponding discriminator of the conditional DCGAN is depicted in Figure 4.10. Note that this project implemented GANs to produce images of dimensions $256 \times 256 \times 3$ and these architectures are for representation purposes. The discriminator and generator models were stacked sequentially so that the images produced by the generator and the original images were fed into the discriminator for classification.

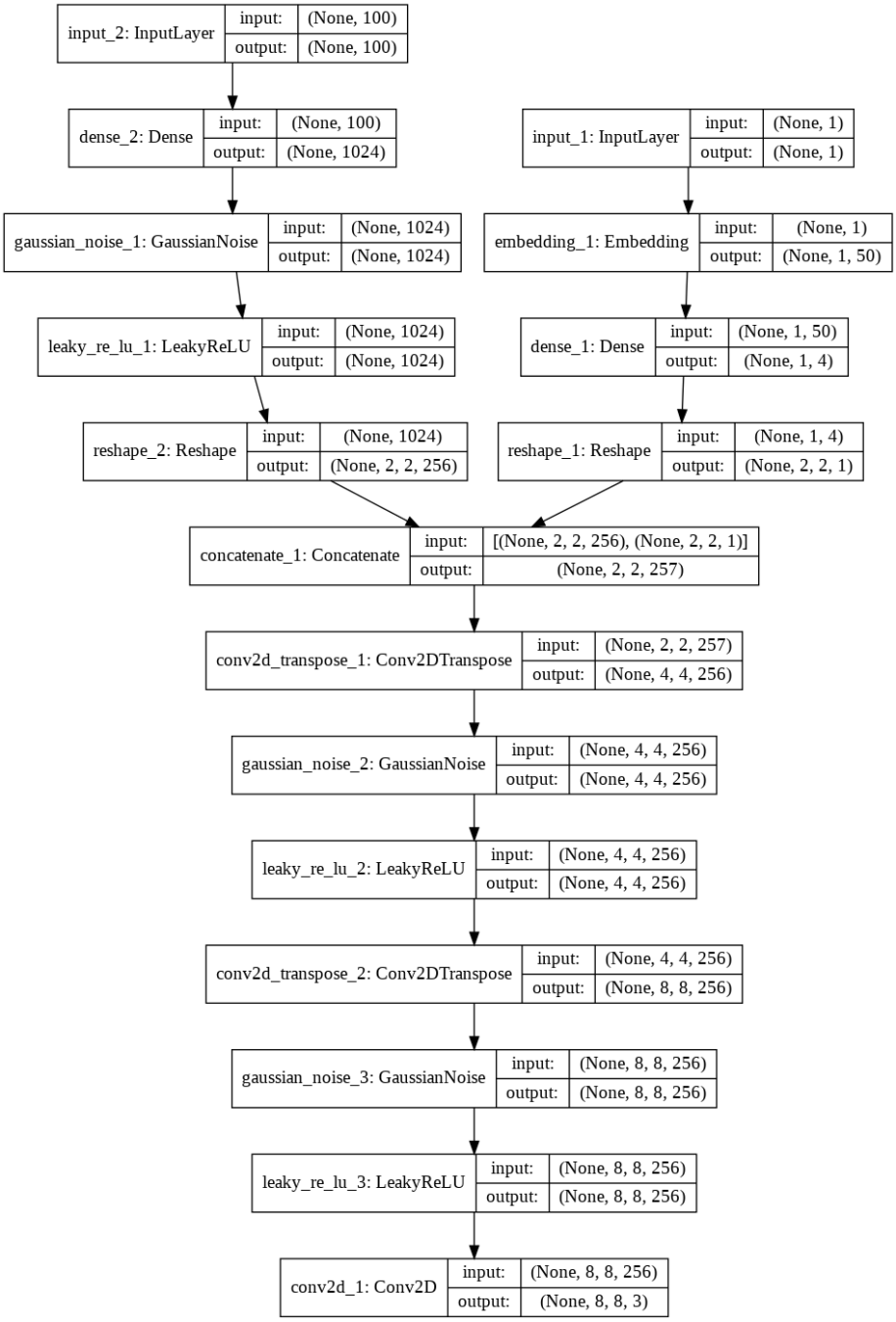

Figure 4.9: Architecture of the Generator of the Conditional DCGAN

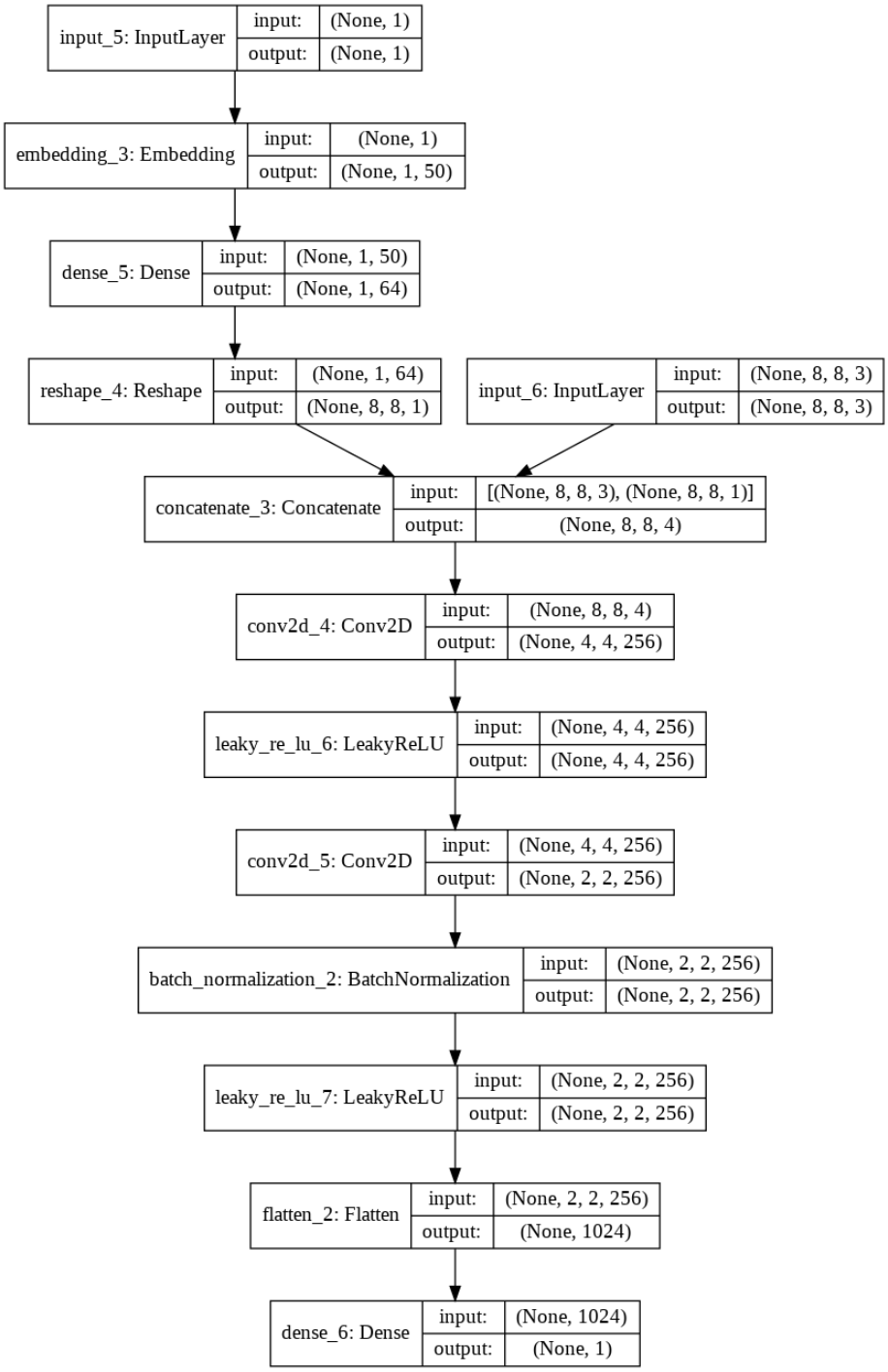

Figure 4.10: Architecture of the Discriminator of the Conditional DCGAN

4.7 Progressively Growing Generative Adversarial Network

The Progressively Growing Generative Adversarial Network (PGGAN) architecture [62] proposed by Karras et. al described an innovative training methodology for GANs in which they propose to train both the generator and the discriminator progressively, starting by training it on images of low resolution and gradually increasing the resolution to later learn on the finer details of the images. This has the major advantage of stabilizing the training process. Further, they implemented minibatch layers to increase variation in the generated images.

In this project, the PGGAN was implemented as an alternative to the DCGAN and was used to generate images as it was expected to generate images having more diversity as well as quality. The same experiments were conducted on these images. However, 12 different models were implemented for generating images of the 12 different viseme classes unlike the DCGAN implementation as the conditional PGGAN model that was implemented failed to capture label information although it generated images of good quality. The architecture of the generator of the PGGAN used to produce images of dimensions $8 \times 8 \times 3$ is depicted in Figure 4.11 and the architecture of the corresponding discriminator is depicted in Figure 4.12. Again, this is for reference only and the images that were generated were of dimensions $256 \times 256 \times 3$.

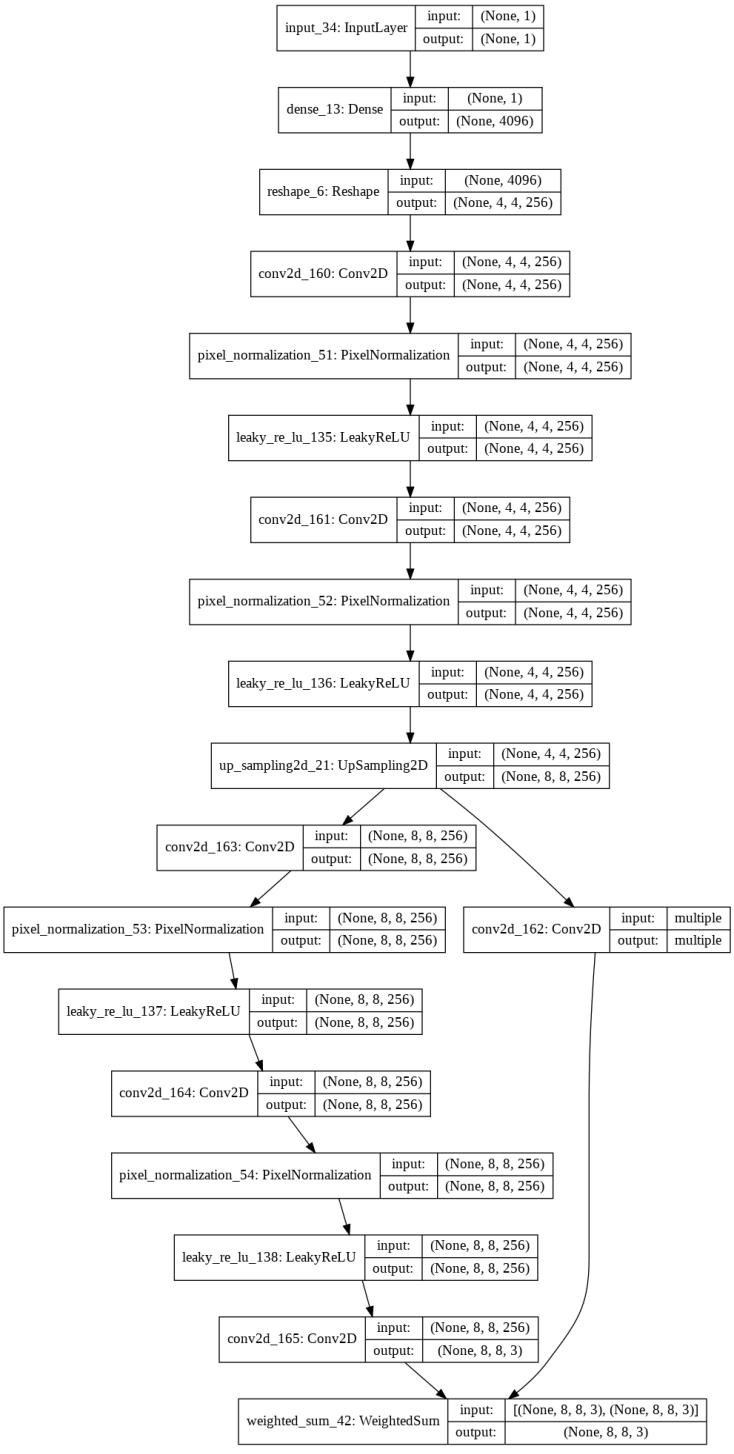

Figure 4.11: Architecture of the Generator of PGGAN

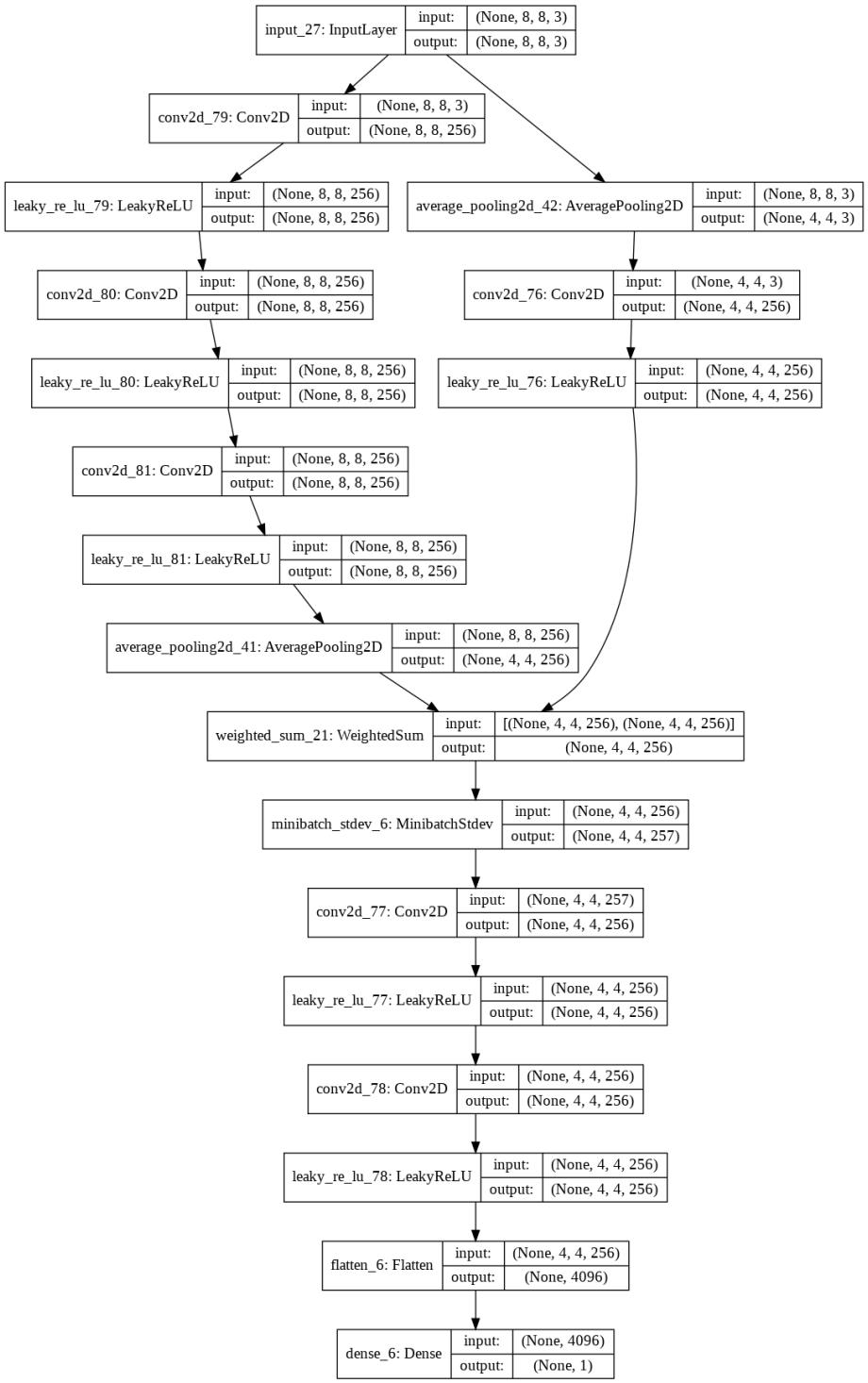

Figure 4.12: Architecture of the Discriminator of PGGAN

Chapter 5

Training and Results

5.1 Classification using VGG16

Classification using VGG16 was done for both phoneme and viseme classes. The optimizer used throughout was the Adam optimizer[63] and both phoneme and viseme-wise classification accuracy was found with a train-test split of 90% - 10%. Dropout rates and filter numbers were varied for optimization. Additionally, an early stopping technique was implemented that depended on the validation loss being lower than the training loss.

Phoneme recognition was done for two different speakers, speaker 01M, a male speaker, and speaker 11F, a female speaker. The learning rate used was 5×10^{-5}. The test accuracy for 01M was found to be 37.57% while it was found to be 35.24% for 11F. Table 5.1 summarizes this:

Table 5.1: Phoneme-wise classification using VGG16

Speaker	**Test Accuracy(%)**
01M	37.57
11F	35.24

The confusion matrices for the predictions on the test dataset for both speakers are shown in Figure 5.1 and 5.2. As each speaker speaks differently, there is a lot of differences between the two. For example, the /z/ phoneme for speaker 01M has been predicted successfully about half the time while the same phoneme has shown poor recognition for 11F.

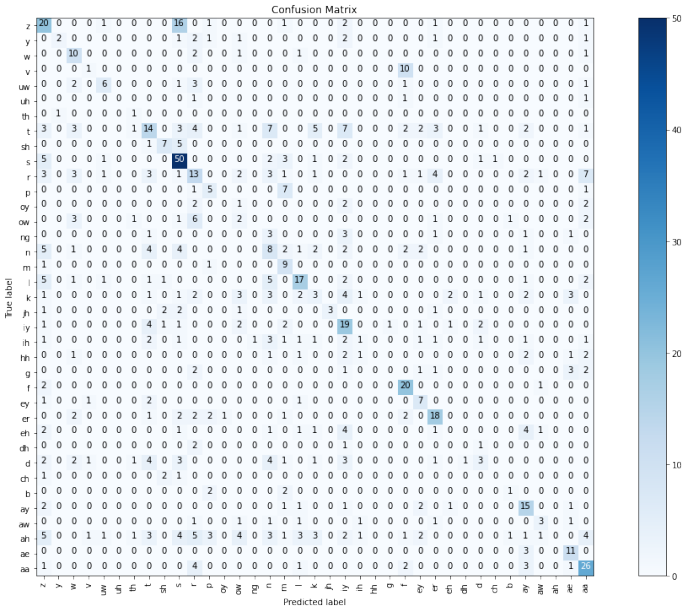

Figure 5.1: 01M Original Images - Phonemes Confusion Matrix

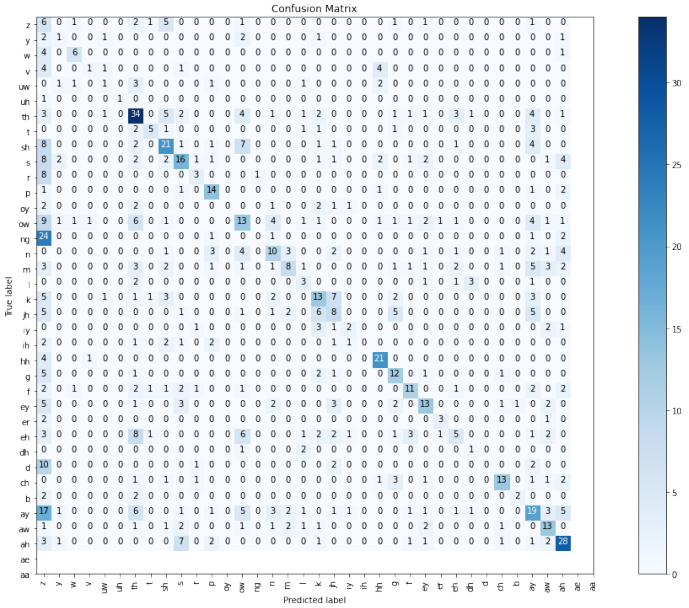

Figure 5.2: 11F Original Images - Phonemes Confusion Matrix

Table 5.2: 01M Original Images - Phonemes Scores

Class	Precision	Recall	F1 Score
z	0.389	0.5	0.438
y	0.5	0.5	0.5
w	0.3	0.429	0.353
v	0.667	0.4	0.5
uw	0.5	0.364	0.421
uh	0	0	0
th	0	0	0
t	0.25	0.24	0.245
sh	0.3	0.333	0.316
s	0.431	0.691	0.531
r	0.25	0.343	0.289
p	0.333	0.556	0.417
oy	0	0	0
ow	0.417	0.5	0.455
ng	0	0	0
n	0.32	0.348	0.333
m	0.34	0.762	0.471
l	0.575	0.5	0.535
k	0.571	0.133	0.216
jh	0	0	0
iy	0.467	0.378	0.418
ih	0	0	0
hh	0.333	0.2	0.25
g	0	0	0
f	0.579	0.917	0.71
ey	0.2	0.167	0.182
er	0.591	0.542	0.565
eh	0.333	0.056	0.095
dh	0	0	0
d	0.444	0.108	0.174
ch	0	0	0
b	0	0	0
ay	0.321	0.474	0.383
aw	0.364	0.5	0.421
ah	0.17	0.209	0.188
ae	0.471	0.444	0.457
aa	0.49	0.521	0.505

Table 5.3: 11F Original Images - Phonemes Scores

Class	Precision	Recall	F1 Score
z	0.4	0.207	0.273
y	0.455	0.714	0.556
w	0.517	0.682	0.588
v	0.125	0.143	0.133
uw	0.429	0.375	0.4
uh	1	0.4	0.571
th	0	0	0
t	0.4	0.473	0.433
sh	0.313	0.455	0.37
s	0.519	0.583	0.549
r	0.421	0.235	0.302
p	0.471	0.5	0.485
oy	0	0	0
ow	0.714	0.417	0.526
ng	1	0.182	0.307
n	0.255	0.463	0.329
m	0.6	0.667	0.631
l	0.25	0.222	0.235
k	0.31	0.281	0.295
jh	0.417	0.555	0.476
iy	0.667	0.339	0.45
ih	0.311	0.368	0.337
hh	1	0.167	0.286
g	0.25	0.333	0.286
f	0.583	0.636	0.609
ey	0.577	0.6	0.588
er	0.408	0.55	0.468
eh	0.194	0.231	0.211
dh	75	0.273	0.4
d	0.25	0.276	0.262
ch	1	0.167	0.286
b	0.5	0.647	0.564
ay	0.6	0.571	0.585
aw	0.5	0.273	0.353
ah	0.26	0.226	0.241
ae	0.656	0.514	0.576
aa	0.464	0.627	0.533

To try and improve the accuracy on the same speaker specific datasets, contours were drawn around the lip images as shown in Figure 5.3:

(a) (b)

Figure 5.3: Pre-processed Images with Contours: (a) Sample Image with Contour for 01M; (b) Sample Image with Contour for 11F

Then, VGG16 was trained again on these images with the same base learning rate. Testing accuracy was found to be 34.51% for 01M and 39.95% for speaker 11F respectively. Below is Table 5.4 that summarizes this:

Table 5.4: Phoneme-wise classification with Contours using VGG16

Speaker	**Test Accuracy(%)**
01M	34.51
11F	39.95

Similarly, viseme recognition was done for the same speakers. The training converged well for 01M with a learning rate of 1×10^{-5} while it converged well for speaker 11F with a learning rate of 5×10^{-5}. The test accuracies of both are summarized in Table 5.5:

Table 5.5: Viseme-wise classification using VGG16

Speaker	**Test Accuracy(%)**
01M	50.31
11F	49.42

Figures 5.4 and 5.5 are the confusion matrices that were obtained for these models. Both show very similar results and both struggle to recognise the visemes V2 and F.

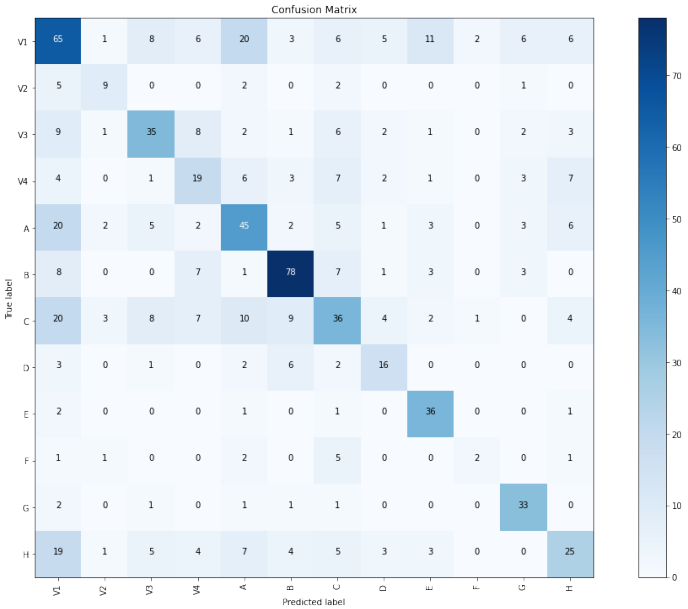

Figure 5.4: 01M Original Images - Visemes Confusion Matrix

Table 5.6: 01M Original Images - Visemes Scores

Class	Precision	Recall	F1 Score
V1	0.272	0.409	0.326
V2	0.555	0.345	0.425
V3	0.515	0.531	0.523
V4	0.484	0.259	0.337
A	0.412	0.339	0.372
B	0.8	0.58	0.673
C	0.343	0.396	0.367
D	0.615	0.64	0.627
E	0.609	0.866	0.716
F	0.273	0.333	0.3
G	0.625	0.714	0.667
H	0.339	0.281	0.308

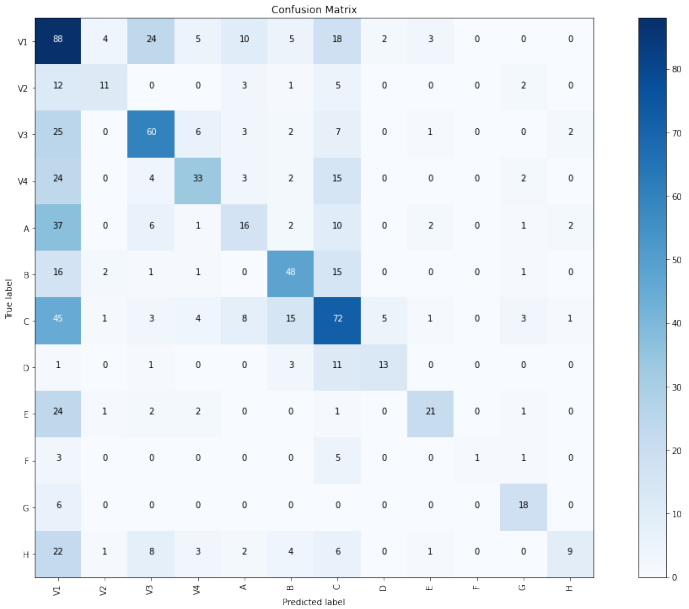

Figure 5.5: 11F Original Images - Visemes Confusion Matrix

Table 5.7: 11F Original Images - Visemes Scores

Class	Precision	Recall	F1 Score
V1	0.456	0.491	0.473
V2	0.475	0.487	0.481
V3	0.567	0.557	0.563
V4	0.51	0.549	0.529
A	0.358	0.232	0.281
B	0.623	0.716	0.667
C	0.492	0.488	0.49
D	0.667	0.692	0.679
E	0.574	0.7	0.63
F	0.5	0.222	0.307
G	0.562	0.78	0.655
H	0.383	0.286	0.327

Again, contour images were used for viseme classification and these results are given in Table 5.8.

Table 5.8: Viseme-wise classification with Contours using VGG16

Speaker	**Test Accuracy(%)**
01M	43.40
11F	46.72

5.2 Utilizing DCGAN for Data Augmentation

With these results established, conditional DCGAN viseme-wise generated images were added to the dataset. For both the speakers, 200 images were generated for each class and then added to the dataset, leading to a total of 2,400 generated images. A sample of the generated images is shown in Figure 5.6.

Figure 5.6: Images of 01M Generated by the Conditional DCGAN

To get an idea of how well the GAN had captured the features of each class, a dataset consisting of only the generated images was fed to VGG16 for classification. A learning rate of 5×10^{-5} for both speakers and the early stopping mechanism described above was used. The results clearly show that the DCGAN has captured the features quite well and are summarized in Table

5.9 below. Note that the same procedure cannot be applied for phoneme-wise classification due to the nature of the mapping between phonemes and visemes.

Table 5.9: Viseme-wise classification of DCGAN Generated Images using VGG16

Speaker	**Test Accuracy(%)**
01M	88.43
11F	91.25

The confusion matrices for these models are presented in Figures 5.7 and 5.8. The classification performance is extremely good for these models and proves definitively that the GAN captures features satisfactorily.

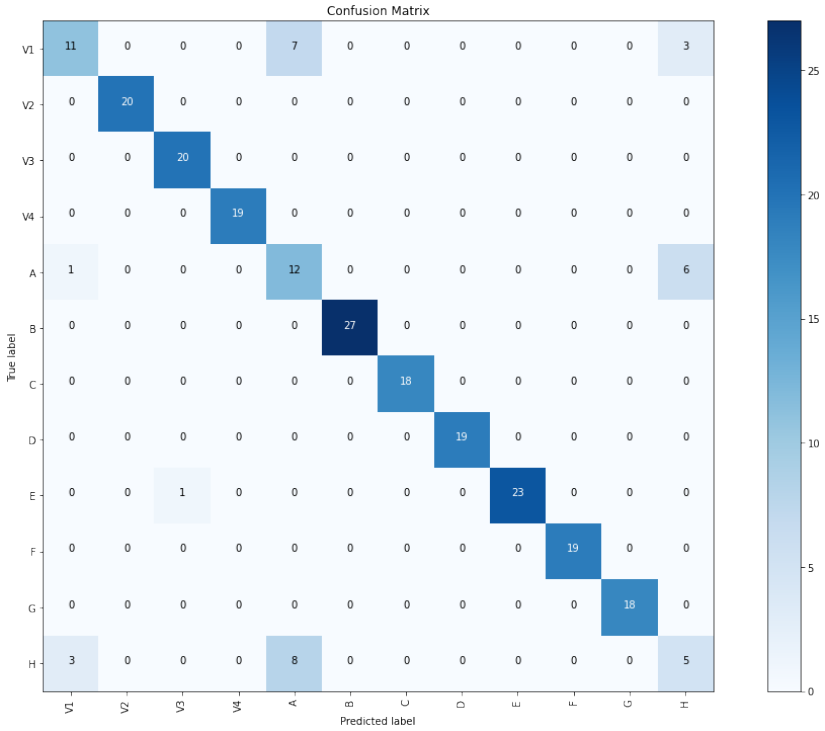

Figure 5.7: 01M DCGAN Generated Images - Visemes Confusion Matrix

Table 5.10: 01M Generated Images - Visemes Scores

Class	Precision	Recall	F1 Score
V1	0.696	0.615	0.653
V2	1	1	1
V3	1	1	1
V4	1	1	1
A	0.75	0.3	0.429
B	1	0.952	0.976
C	1	1	1
D	0.95	1	0.974
E	1	1	1
F	1	1	1
G	1	1	1
H	0.382	0.684	0.491

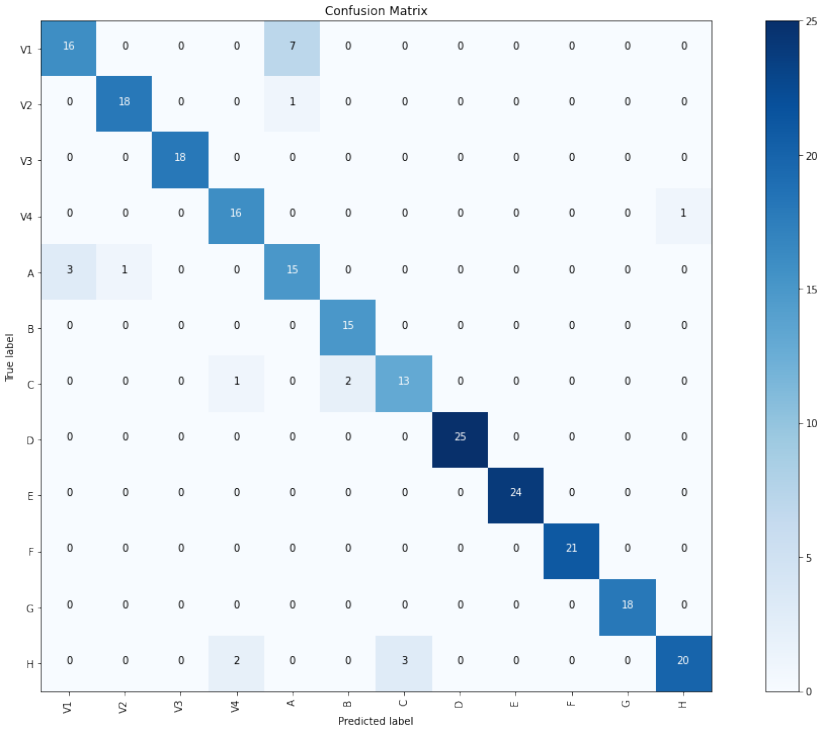

Figure 5.8: 11F DCGAN Generated Images - Visemes Confusion Matrix

Table 5.11: 11F Generated Images - Visemes Scores

Class	Precision	Recall	F1 Score
V1	0.75	0.857	0.8
V2	1	0.722	0.839
V3	1	1	1
V4	0.857	0.947	0.9
A	0.72	0.818	0.766
B	0.905	1	0.95
C	1	0.882	0.938
D	1	1	1
E	1	1	1
F	1	1	1
G	1	1	1
H	0.95	0.864	0.905

After this, the real and generated images were combined to form a single dataset and viseme-wise classification was found out using VGG16 in two different ways:

1. In type 1, the dataset containing both the real and generated images was shuffled and later, the images were split into training and testing data. Therefore, both testing and training data had real and generated images. For this dataset, the classification accuracy increased considerably to 59.21% for 01M and 55.48% for 11F. The learning rate used was 5×10^{-5}. The results are summarized in Table 5.12 below:

Table 5.12: Type 1 - Viseme-wise classification of Combined Images using VGG16

Speaker	**Test Accuracy(%)**
01M	59.21
11F	55.48

The confusion matrices for these models are presented in Figures 5.9 and 5.10. There is a significant increase in performance, especially in recognition of the visemes F and V2.

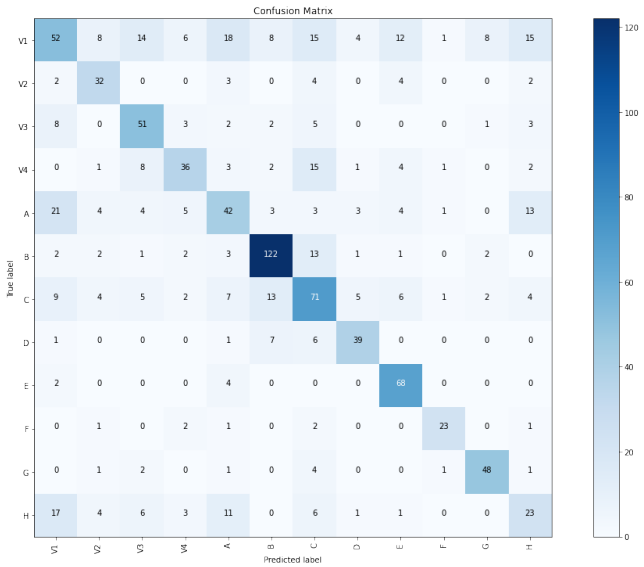

Figure 5.9: 01M Type 1 Approach - Visemes Confusion Matrix

Table 5.13: 01M Type 1 Approach - Visemes Scores

Class	Precision	Recall	F1 Score
V1	0.315	0.544	0.399
V2	0.614	0.643	0.628
V3	0.754	0.561	0.643
V4	0.422	0.302	0.352
A	0.6	0.305	0.404
B	0.752	0.734	0.743
C	0.642	0.464	0.538
D	0.709	0.886	0.788
E	0.652	0.87	0.745
F	0.692	0.6	0.643
G	0.627	0.914	0.744
H	0.406	0.341	0.371

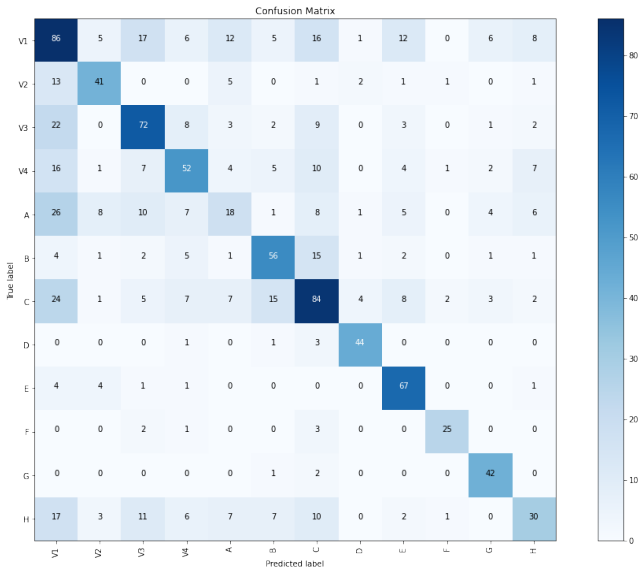

Figure 5.10: 11F Type 1 Approach - Visemes Confusion Matrix

Table 5.14: 11F Type 1 Approach - Visemes Scores

Class	Precision	Recall	F1 Score
V1	0.452	0.54	0.492
V2	0.62	0.667	0.643
V3	0.616	0.699	0.655
V4	0.476	0.482	0.479
A	0.411	0.289	0.339
B	0.641	0.667	0.653
C	0.488	0.477	0.483
D	0.81	0.732	0.769
E	0.67	0.732	0.7
F	0.93	0.844	0.885
G	0.744	0.854	0.795
H	0.638	0.353	0.454

2. In type 2, testing data was taken only from the original dataset and then, the generated images were added to the training dataset. Therefore, the training dataset was augmented using the generated images while the test dataset was made up of real images only. The results for this dataset configuration are displayed in Table 5.15 below:

   Clearly, there is a decrease in classification accuracy compared to type 1 for both speakers although speaker 01M's decrease is considerable. This

Table 5.15: Type 2 - Viseme-wise classification of Combined Images using VGG16

Speaker	**Test Accuracy(%)**
01M	51.33
11F	50.00

might have happened because the generated images were not of the same quality as the original images and has to be explored further. Also, note that type 2 leads to a negligible increase in accuracy when compared to the performance obtained only on the original dataset. The confusion matrices for these models are presented in Figures 5.11 and 5.12.

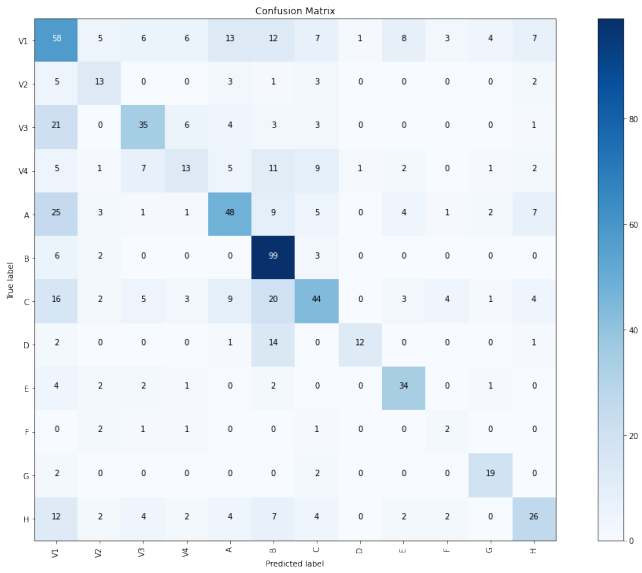

Figure 5.11: 01M Type 2 Approach - Visemes Confusion Matrix

Table 5.16: 01M Type 2 Approach - Visemes Scores

Class	Precision	Recall	F1 Score
V1	0.351	0.575	0.436
V2	0.333	0.217	0.263
V3	0.582	0.485	0.529
V4	0.518	0.468	0.492
A	0.492	0.352	0.411
B	0.636	0.701	0.667
C	0.509	0.463	0.485
D	0.692	0.545	0.61
E	0.6	0.792	0.683
F	0.167	0.125	0.143
G	0.688	0.647	0.667
H	0.458	0.18	0.259

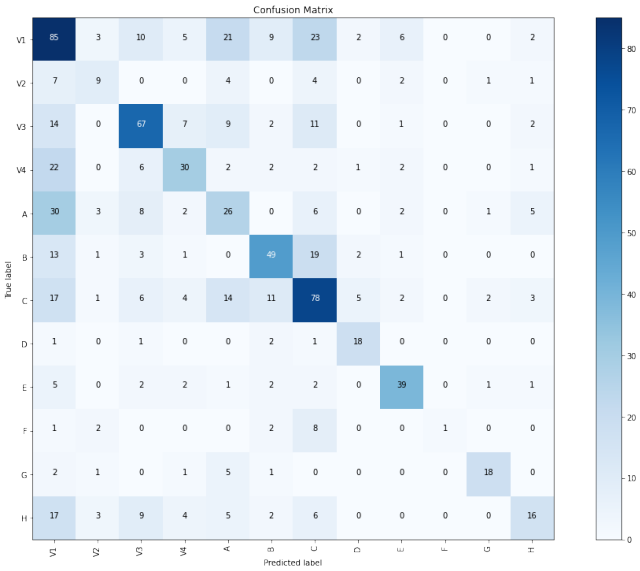

Figure 5.12: 11F Type 2 Approach - Visemes Confusion Matrix

Table 5.17: 11F Type 2 Approach - Visemes Scores

Class	Precision	Recall	F1 Score
V1	0.354	0.598	0.445
V2	0.485	0.447	0.466
V3	0.528	0.528	0.528
V4	0.557	0.403	0.468
A	0.42	0.236	0.302
B	0.625	0.461	0.53
C	0.5	0.436	0.466
D	0.667	0.667	0.667
E	0.589	0.793	0.676
F	0.6	0.273	0.375
G	0.617	0.75	0.677
H	0.3	0.13	0.182

5.3 Utilizing PGGAN for Data Augmentation

Once again, all the experiments that were conducted using the DCGAN generated images were done using the PGGAN generated images. A sample of the images generated by the different PGGANs is shown in Figure 5.13:

Figure 5.13: Images of 01M Generated by the PGGANs

Again, to understand how well the PGGAN had captured the features of each class, classification was done only on these generated images, the results for which are displayed in Table 5.18.

Table 5.18: Viseme classification of PGGAN Generated Images using VGG16

Speaker	**Test Accuracy(%)**
01M	100.00
11F	100.00

The confusion matrices for these models are presented in Figures 5.14 and 5.15. Clearly, the PGGAN does an even better job than the DCGAN as it captures features with 100% accuracy.

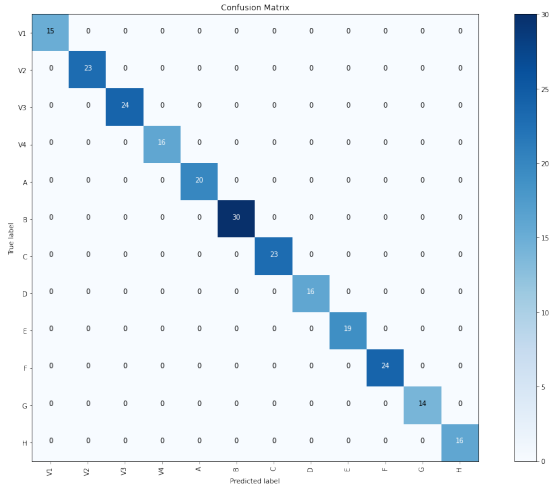

Figure 5.14: 01M PGGAN Generated Images - Visemes Confusion Matrix

Table 5.19: 01M Generated Images PGGAN - Visemes Scores

Class	Precision	Recall	F1 Score
V1	1.0	1.0	1.0
V2	1.0	1.0	1.0
V3	1.0	1.0	1.0
V4	1.0	1.0	1.0
A	1.0	1.0	1.0
B	1.0	1.0	1.0
C	1.0	1.0	1.0
D	1.0	1.0	1.0
E	1.0	1.0	1.0
F	1.0	1.0	1.0
G	1.0	1.0	1.0
H	1.0	1.0	1.0

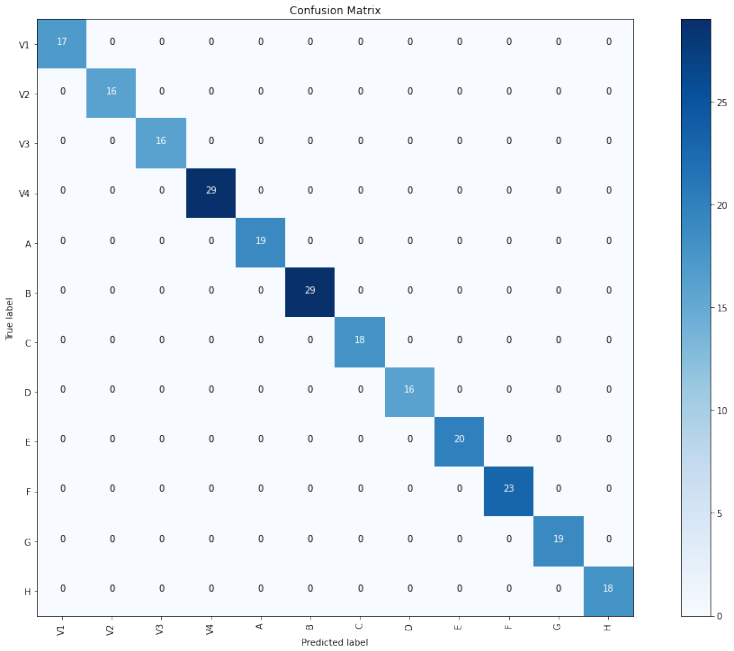

Figure 5.15: 11F PGGAN Generated Images - Visemes Confusion Matrix

Table 5.20: 11F Generated Images PGGAN - Visemes Scores

Class	Precision	Recall	F1 Score
V1	1.0	1.0	1.0
V2	1.0	1.0	1.0
V3	1.0	1.0	1.0
V4	1.0	1.0	1.0
A	1.0	1.0	1.0
B	1.0	1.0	1.0
C	1.0	1.0	1.0
D	1.0	1.0	1.0
E	1.0	1.0	1.0
F	1.0	1.0	1.0
G	1.0	1.0	1.0
H	1.0	1.0	1.0

The experiments conducted using the images generated by the DCGAN was again implemented using the PGGAN.

1. The Type 1 approach showed even higher accuracies as shown in Table 5.21.

Table 5.21: Type 1 PGGAN - Viseme classification of Combined Images using VGG16

Speaker	**Test Accuracy(%)**
01M	62.63
11F	60.52

The confusion matrices for these models are presented in Figures 5.16 and 5.17.

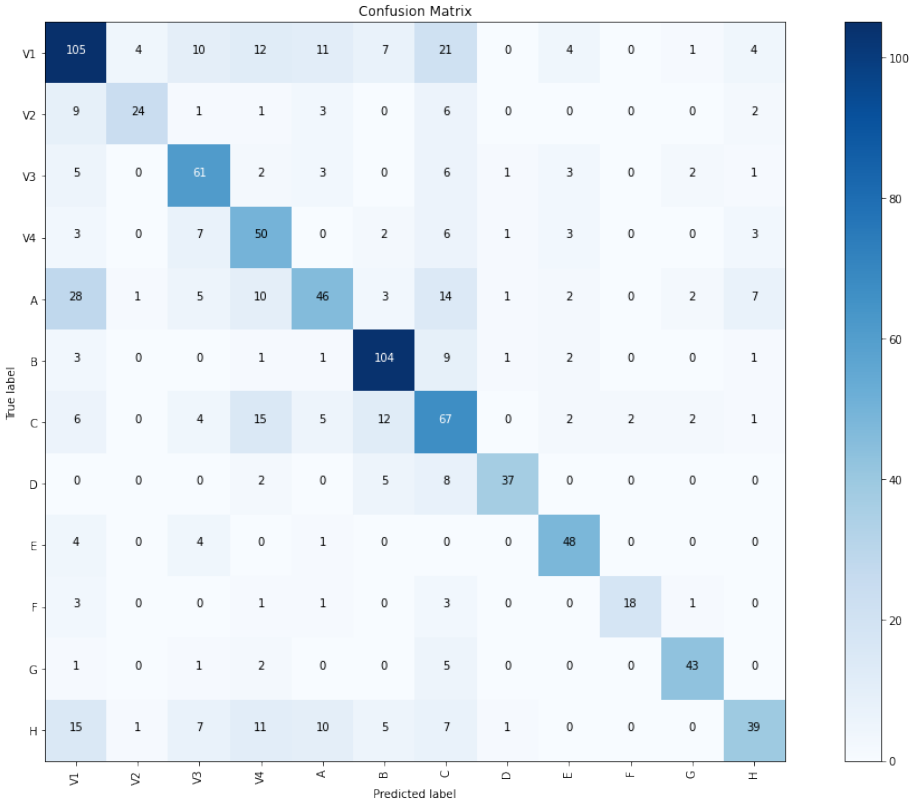

Figure 5.16: 01M Type 1 Approach PGGAN - Visemes Confusion Matrix

Table 5.22: 01M Type 1 Approach PGGAN - Visemes Scores

Class	Precision	Recall	F1 Score
V1	0.577	0.587	0.582
V2	0.8	0.522	0.632
V3	0.61	0.726	0.663
V4	0.467	0.667	0.549
A	0.568	0.387	0.46
B	0.754	0.852	0.8
C	0.441	0.578	0.5
D	0.881	0.712	0.787
E	0.75	0.842	0.793
F	0.9	0.667	0.766
G	0.843	0.827	0.835
H	0.672	0.406	0.506

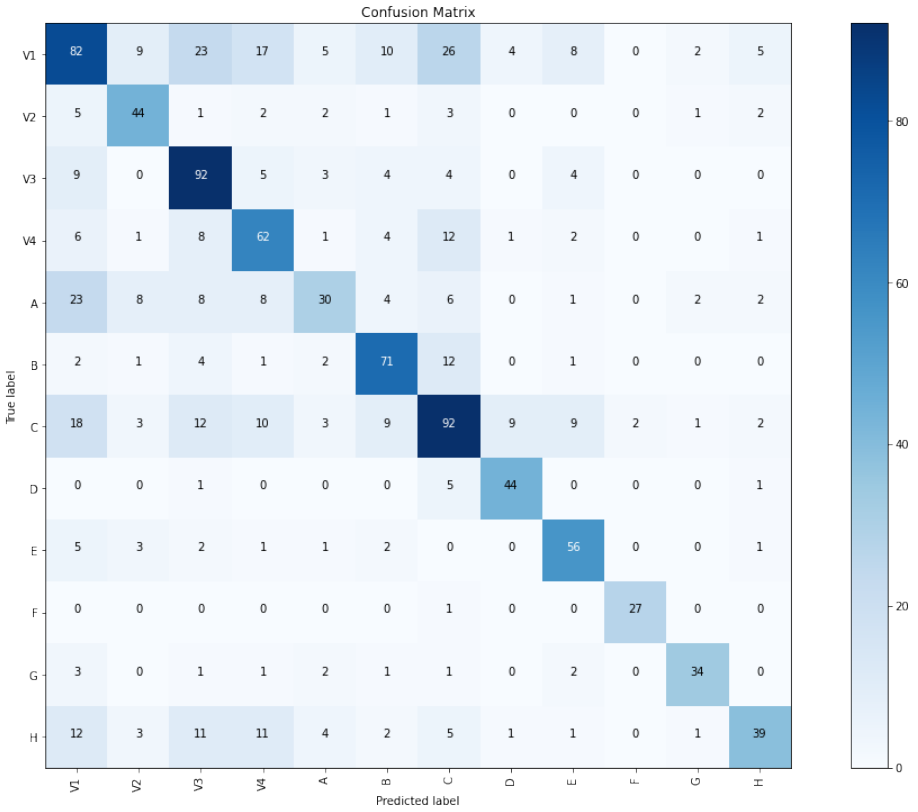

Figure 5.17: 11F Type 1 Approach PGGAN - Visemes Confusion Matrix

Table 5.23: 11F Type 1 Approach PGGAN - Visemes Scores

Class	Precision	Recall	F1 Score
V1	0.497	0.429	0.461
V2	0.611	0.721	0.662
V3	0.564	0.76	0.648
V4	0.525	0.633	0.574
A	0.566	0.326	0.414
B	0.657	0.755	0.703
C	0.551	0.541	0.546
D	0.746	0.863	0.8
E	0.667	0.789	0.723
F	0.931	0.964	0.947
G	0.829	0.756	0.791
H	0.736	0.433	0.545

2. The Type 2 approach using PGGAN images also showed a marked increase in accuracy compared to the same that used DCGAN images as shown in table 5.24.

Table 5.24: Type 2 PGGAN - Viseme classification of Combined Images using VGG16

Speaker	**Test Accuracy(%)**
01M	53.24
11F	51.61

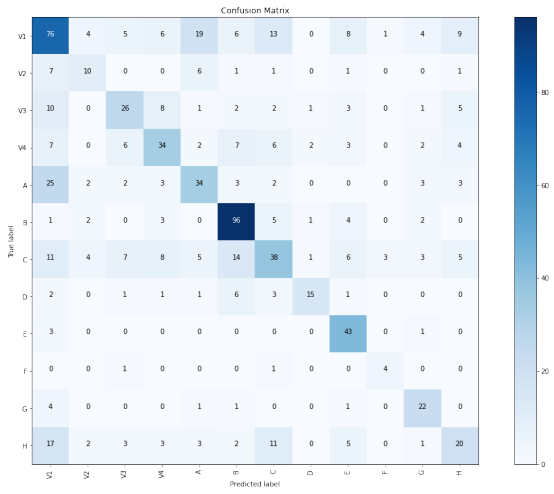

Figure 5.18: 01M Type 2 Approach PGGAN - Visemes Confusion Matrix

Table 5.25: 01M Type 2 Approach PGGAN - Visemes Scores

Class	Precision	Recall	F1 Score
V1	0.466	0.503	0.484
V2	0.417	0.37	0.392
V3	0.51	0.441	0.473
V4	0.515	0.466	0.489
A	0.472	0.442	0.456
B	0.696	0.842	0.762
C	0.463	0.362	0.406
D	0.75	0.5	0.6
E	0.573	0.915	0.705
F	0.5	0.667	0.571
G	0.564	0.759	0.647
H	0.426	0.299	0.351

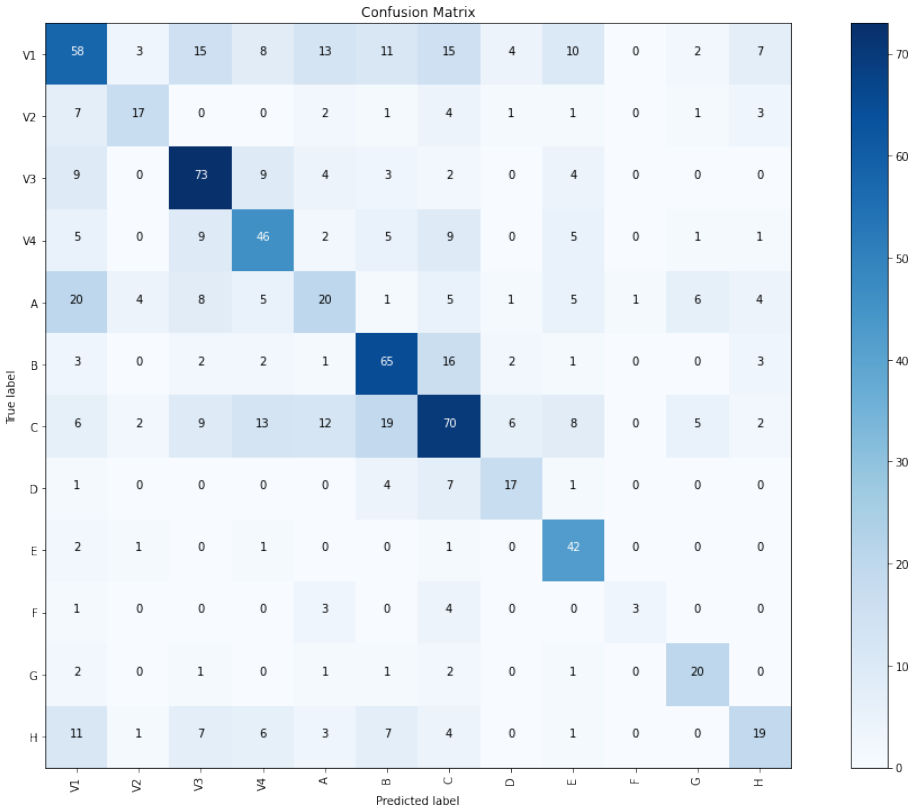

Figure 5.19: 11F Type 2 Approach PGGAN - Visemes Confusion Matrix

Table 5.26: 11F Type 2 Approach PGGAN - Visemes Scores

Class	Precision	Recall	F1 Score
V1	0.497	0.429	0.461
V2	0.611	0.721	0.662
V3	0.564	0.76	0.648
V4	0.525	0.633	0.574
A	0.566	0.326	0.414
B	0.657	0.755	0.703
C	0.551	0.541	0.546
D	0.746	0.863	0.8
E	0.667	0.789	0.723
F	0.931	0.964	0.947
G	0.829	0.756	0.791
H	0.736	0.433	0.545

5.4 Classification Using AlexNet

For classification using AlexNet, stochastic gradient descent was used as the optimiser. A total of 121,041 images from 15 different speakers was used for the speaker independent phoneme classification scenario, which did not include the phoneme -/sil/ (silence) and out of these 121,041 images, 90% of images were used for training and the remaining 10% for validation and testing. The maximum testing accuracy achieved was 37.92% and the corresponding base learning rate was 0.0001.

Phoneme classification was also done on two different speakers individually, namely the male speaker 01M and the female speaker 11F. While 11F had 8,710 images, speaker 01M had 7,845 images. Here as well, 90% of the images was used for training and 10% for validation and testing.The testing accuracies for speaker 01M and 11F were 35.78% and 24.13% respectively. Below is Table 5.27 that summarizes this:

Table 5.27: Phoneme-wise classification using AlexNet

Speaker	**Test Accuracy(%)**
Independent	37.92
01M	35.78
11F	24.13

The confusion matrices for these models that classified between phonemes is given in Figures 5.20 and 5.21. It is clearly seen that for both speakers, many images belonging to phoneme /z/ is being misclassified as phoneme /s/. This was anticipated as both these phonemes have the same viseme.

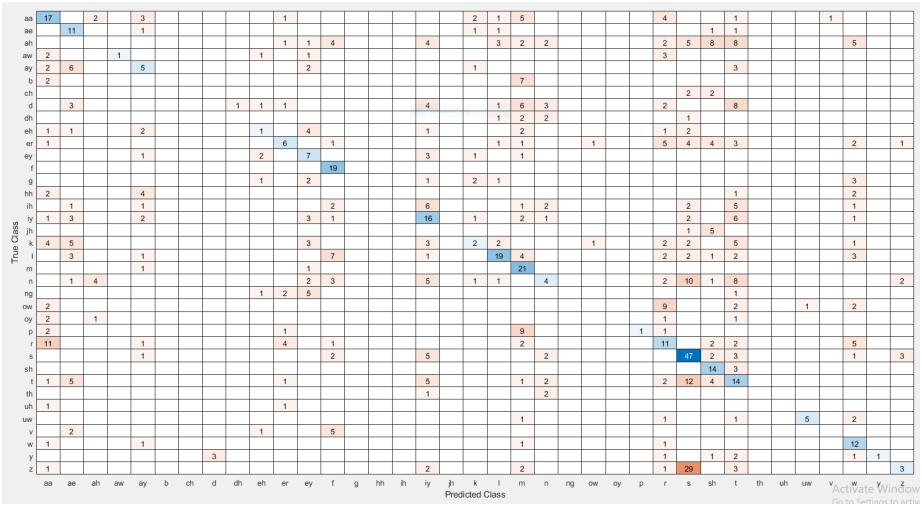

Figure 5.20: 01M Phoneme Confusion Matrix

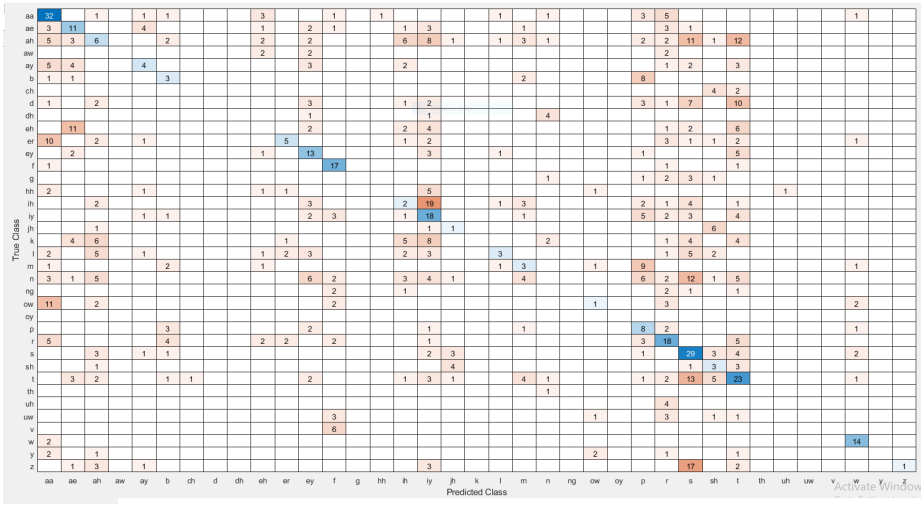

Figure 5.21: 11F Phoneme Confusion Matrix

Table 5.28: 01M Original Images AlexNet - Phonemes Scores

Class	Precision	Recall	F1 Score
z	0.073	0.333	0.12
y	0.111	0.5	0.182
w	0.75	0.3	0.429
v	0.0	0.0	0
uw	0.5	0.833	0.625
uh	0.0	0	0
th	0.0	0	0
t	0.298	0.189	0.231
sh	0.824	0.264	0.4
s	0.712	0.42	0.528
r	0.282	0.186	0.224
p	0.071	0.333	0.118
oy	0.0	0	0
ow	0.0	0.0	0
ng	0.0	0	0
n	0.091	0.2	0.125
m	0.913	0.3	0.452
l	0.422	0.613	0.5
k	0.067	0.182	0.098
jh	0.0	0	0
iy	0.41	0.281	0.333
ih	0.0	0	0
hh	0.0	0	0
g	0.0	0	0
f	1.0	0.38	0.551
ey	0.467	0.241	0.318
er	0.2	0.375	0.261
eh	0.067	0.143	0.091
dh	0.0	0.0	0
d	0.0	0.0	0
ch	0.0	0	0
b	0.0	0	0
ay	0.263	0.208	0.233
aw	0.125	1.0	0.222
ah	0.0	0.0	0
ae	0.688	0.268	0.386
aa	0.459	0.321	0.378

Table 5.29: 11F Original Images Alexnet - Phonemes Scores

Class	Precision	Recall	F1 Score
z	0.036	1.0	0.069
y	0.0	0	0
w	0.875	0.636	0.737
v	0.0	0.0	0
uw	0.0	0	0
uh	0.0	0.0	0
th	0.0	0	0
t	0.359	0.258	0.301
sh	0.25	0.107	0.15
s	0.592	0.248	0.349
r	0.429	0.286	0.343
p	0.444	0.151	0.225
oy	0	0	0
ow	0.048	0.167	0.074
ng	0.0	0	0
n	0.0	0.0	0
m	0.158	0.115	0.133
l	0.1	0.333	0.154
k	0.0	0	0
jh	0.111	0.091	0.1
iy	0.439	0.207	0.281
ih	0.053	0.077	0.062
hh	0.0	0.0	0
g	0.0	0	0
f	0.85	0.436	0.576
ey	0.5	0.283	0.361
er	0.172	0.455	0.25
eh	0.05	0.067	0.057
dh	0.0	0	0
d	0.0	0	0
ch	0.0	0.0	0
b	0.2	0.167	0.182
ay	0.167	0.267	0.205
aw	0.0	0	0
ah	0.088	0.143	0.109
ae	0.355	0.268	0.306
aa	0.627	0.372	0.467

Then, AlexNet was trained again on the contour images with the same base learning rate of 0.0001. Testing accuracy was found to be 37.82% for 01M and 36.45% for speaker 11F respectively. Below is Table 5.30 that summarizes this:

Table 5.30: Phoneme-wise classification with Contours using AlexNet

Speaker	**Test Accuracy(%)**
01M	37.82
11F	36.45

Next, AlexNet was trained for viseme classification. The optimiser used was the Adam optimizer while the base learning rate was 0.001. Here again, 90% of the images was used for training while the rest was used for validation and testing.The test accuracy was found to be 43.12% for speaker independent classification. For the speaker specific scenario, the base learning rate used was 0.0001. Similar to speaker independent training, 90% of the dataset was used for training with the remaining 10% used for validation and testing. The testing accuracies for 01M and 11F were found to be 47.04% and 45.55% respectively. Below is Table 5.31 that summarizes this:

Table 5.31: Viseme-wise classification using AlexNet

Speaker	**Test Accuracy(%)**
Independent	43.12
01M	47.04
11F	45.55

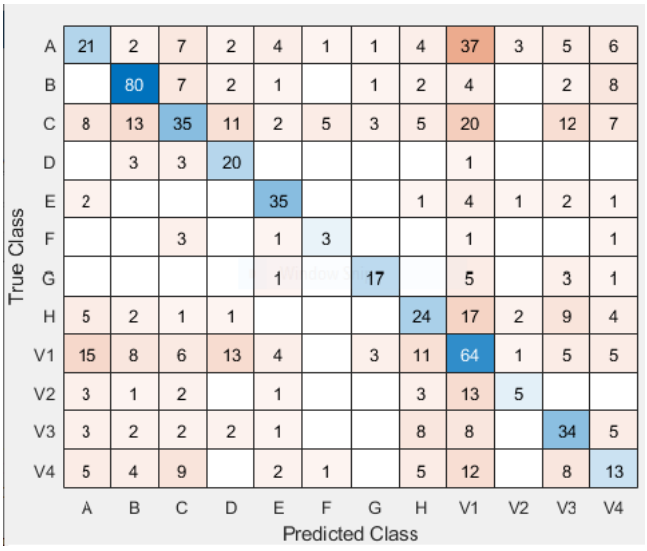

Figure 5.22: 01M Viseme Confusion Matrix

Table 5.32: 01M CNN - Visemes Scores

Class	Precision	Recall	F1 Score
V1	0.761	0.673	0.714
V2	0.333	0.3	0.316
V3	0.63	0.68	0.654
V4	0.369	0.381	0.375
A	0.474	0.344	0.399
B	0.179	0.417	0.25
C	0.523	0.425	0.469
D	0.22	0.255	0.236
E	0.226	0.339	0.271
F	0.748	0.696	0.721
G	0.289	0.467	0.357
H	0.741	0.392	0.513

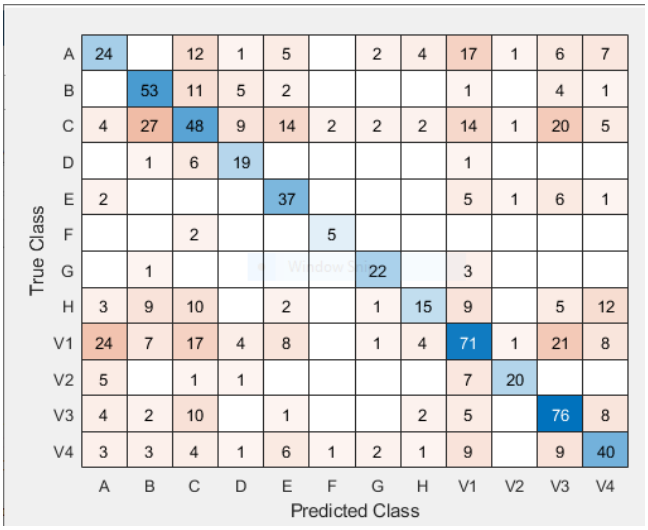

Figure 5.23: 11F Viseme Confusion Matrix

From the viseme confusion matrices shown in Figures 5.22 and 5.23, it is seen that a lot of images of the viseme class 'A' are misclassified as viseme 'V1'.

Table 5.33: 11F CNN - Visemes Scores

Class	Precision	Recall	F1 Score
V1	0.712	0.493	0.583
V2	0.714	0.625	0.667
V3	0.846	0.733	0.786
V4	0.227	0.536	0.319
A	0.452	0.5	0.475
B	0.588	0.833	0.69
C	0.704	0.551	0.618
D	0.506	0.488	0.497
E	0.304	0.348	0.324
F	0.688	0.515	0.589
G	0.324	0.397	0.357
H	0.704	0.475	0.567

Again, AlexNet was trained on contour images for viseme classification the details of which is given below:

Table 5.34: Viseme-wise classification with Contours using AlexNet

Speaker	**Test Accuracy(%)**
01M	45.95
11F	45.91

5.5 Classification using AlexNet CNN-LSTM Hybrid Model

The CNN-LSTM hybrid model was trained for phoneme as well as viseme classification with AlexNet used for feature extraction and LSTM for classification. The hybrid model was trained on the speakers 01M and 11F. 01M had 3141 videos that were reconstructed as described in section 3.2 while speaker 11F had 3263 videos. Similar to the classification process used for image frames, 90% of the videos was used for training and the rest for testing and validation. The number of nodes of LSTM layers and dropout was varied and the best testing accuracy was found to be 38.41% for phoneme classification. Note that generated images were not used for these models as we could not generate images that were temporally related. The different configurations of the LSTM network used in phoneme classification for speaker 01M is summarised in Table 5.35.

Table 5.35: Phoneme-wise classification for 01M Using CNN-LSTM

Number of cells for LSTM Layers 1, 2	**Dropout after LSTM Layers 1, 2**	**Test Accuracy(%)**
516, 256	0.6, 0.4	38.41
1000, 500	0.75, 0.25	32.70
1024, 516	0.6, 0.4	33.65
1024, 2048	0.4, 0.6	37.46

The different configurations of the LSTM network used in phoneme classification for speaker 11F is summarised in Table 5.36.:

Table 5.36: Phoneme-wise classification for 11F Using CNN-LSTM

Number of cells for LSTM Layers 1, 2	**Dropout after LSTM Layers 1, 2**	**Test Accuracy(%)**
516, 256	0.6, 0.4	29.36
1000, 500	0.75, 0.25	33.64
1024, 516	0.6, 0.4	33.33
1024, 2048	0.75, 0.25	36.09

Similar to phoneme classification, viseme classification was done for different configurations of the LSTM layers and the maximum testing accuracy for speaker 01M was found to be 57.14%. The maximum testing accuracy for speaker 11F was 53.4%. The different configurations of the LSTM network used in viseme classification for speaker 01M is summarised in Table 5.37 below.

Table 5.37: Viseme-wise classification for 01M Using CNN-LSTM

Number of cells for LSTM Layers 1, 2	**Dropout after LSTM Layers 1, 2**	**Testing Accuracy(%)**
1024, 516	0.75, 0.25	50.00
1024, 2048	0.75, 0.25	57.14
1024, 2048	0.4, 0.6	47.14
2048, 4096	0.75, 0.25	47.14

The confusion matrix for the best 01M model is given in Figure 5.24:

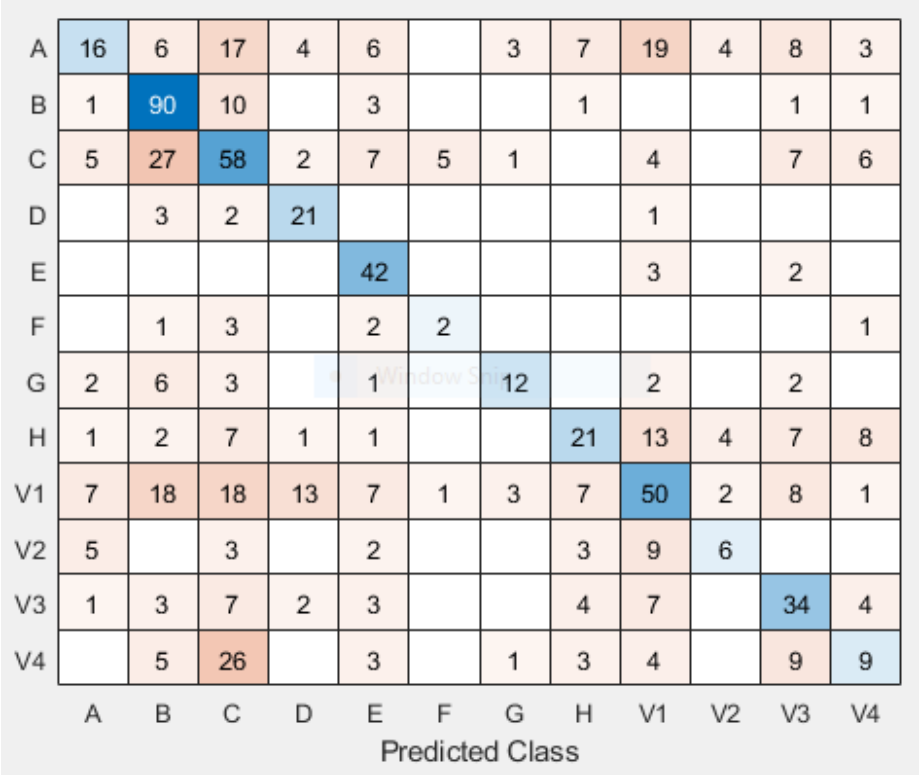

Figure 5.24: 01M Best CNN-LSTM Model Viseme Confusion Matrix

Table 5.38: 01M CNN-LSTM - Visemes Scores

Class	Precision	Recall	F1 Score
V1	0.875	0.545	0.672
V2	0.222	0.25	0.235
V3	0.429	0.6	0.5
V4	0.323	0.457	0.378
A	0.37	0.446	0.405
B	0.214	0.375	0.273
C	0.523	0.436	0.476
D	0.15	0.265	0.191
E	0.172	0.421	0.244
F	0.841	0.559	0.672
G	0.475	0.377	0.42
H	0.778	0.488	0.6

The different configurations of the LSTM network used in viseme classifica-

tion for speaker 11F is summarised in Table 5.39 below:

Table 5.39: Viseme-wise classification for 11F Using CNN-LSTM

Number of cells for LSTM Layers 1, 2	**Dropout after LSTM Layers 1, 2**	**Test Accuracy(%)**
1024, 516	0.75, 0.25	51.73
1024, 2048	0.75, 0.25	53.40
1024, 2048	0.4, 0.6	46.80
2048, 4096	0.75, 0.25	49.13

The confusion matrix for the best 11F model is given in Figure 5.25:

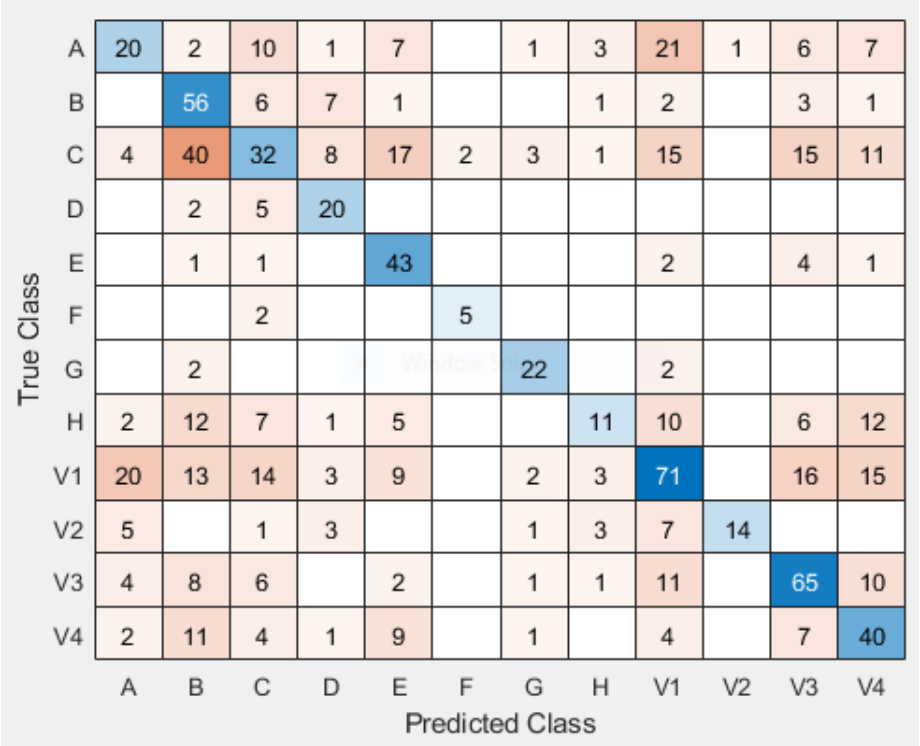

Figure 5.25: 11F Best CNN-LSTM Model Viseme Confusion Matrix

From the viseme confusion matrices of the CNN-LSTM models, it can be observed that the mis-classification of viseme 'A' as 'V1' is reduced but the viseme 'C' being mis-classified as 'B' has increased for the both the speakers. .

Table 5.40: 11F CNN-LSTM - Visemes Scores

Class	Precision	Recall	F1 Score
V1	0.827	0.462	0.593
V2	0.714	0.714	0.714
V3	0.846	0.71	0.772
V4	0.167	0.478	0.247
A	0.428	0.49	0.457
B	0.412	0.933	0.571
C	0.602	0.533	0.565
D	0.506	0.412	0.455
E	0.253	0.351	0.294
F	0.727	0.381	0.5
G	0.216	0.364	0.271
H	0.741	0.455	0.563

5.6 Classification using VGG16 CNN-HMM Hybrid Model

The CNN-HMM model was trained on both phonemes as well as visemes with VGG16 used for feature extraction and HMM for classification. The model was trained on speakers 01M and 11F. The images were segregated into folders as described in section 4.4. 90% of the dataset was used for training and the rest for testing and validation. Note that generated images were not used for these models as we could not generate images that were temporally related. A feature vector is extracted from each image using VGG16. Feature vectors are taken from all the images that form a phoneme and the vectors are appended to form a linear array of dimension $N \times 1$. This linear sequence is fed into the HMM model for training. The same procedure is followed for testing and log probability is calculated for the predicted phoneme. The Baum-Welch algorithm is used for training and the Viterbi algorithm is used for testing the HMM models. The number of states(n_components) of the HMM was varied and the best testing accuracy for phoneme classification was found to be 32.41% for 01M and 30.043% for 11F. This is summarized in Table 5.41 below:

Table 5.41: Phoneme-wise classification using CNN-HMM

Speaker	**N_Components(States) at Max Accuracy**	**Test Accuracy(%)**
01M	4	32.41
11F	5	30.043

The confusion matrices for these models are given below:

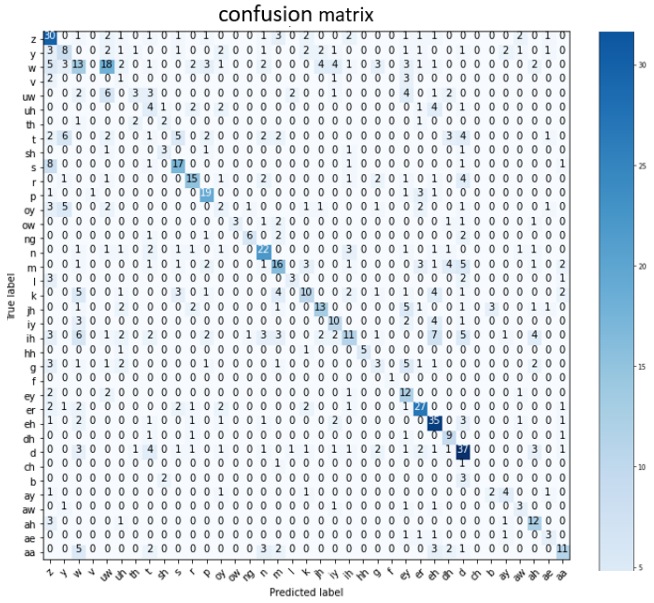

Figure 5.26: 01M CNN-HMM Phoneme Confusion Matrix

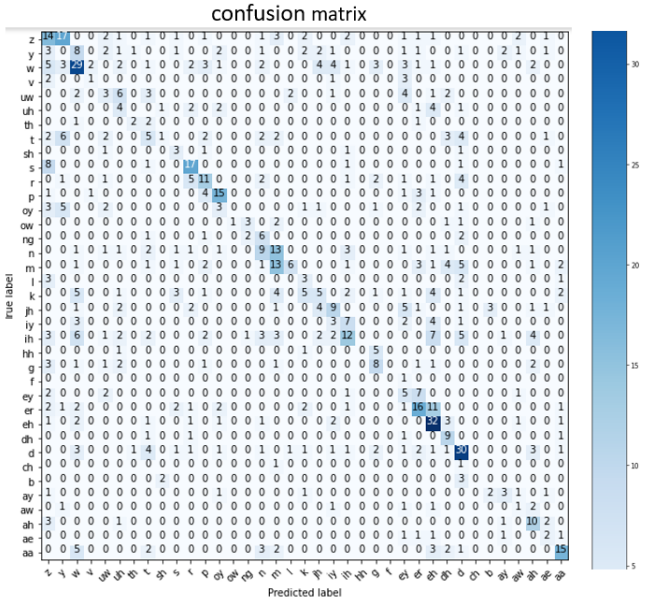

Figure 5.27: 11F CNN-HMM Phoneme Confusion Matrix

Table 5.42: 01M CNN-HMM - Phonemes Scores

Class	Precision	Recall	F1 Score
z	0.42	0.58	0.487
y	0.32	0.26	0.286
w	0.27	0.19	0.23
v	0.0	0.0	0.0
uw	0.15	0.25	0.2
uh	0.0	0.0	0.0
th	0.28	0.33	0.3
t	0.043	0.04	0.08
sh	0.375	0.42	0.39
s	0.53	0.59	0.56
r	0.577	0.52	0.547
p	0.575	0.73	0.64
oy	0.142	0.1	0.12
ow	1.0	0.33	0.496
ng	0.75	0.5	0.60
n	0.594	0.57	0.58
m	0.457	0.38	0.41
l	0.5	0.33	0.0.375
k	0.434	0.28	0.34
jh	0.59	0.419	0.52
iy	0.434	0.5	0.46
ih	0.423	0.196	0.27
hh	1.0	0.714	0.83
g	0.23	0.142	0.175
f	1.0	1.0	1.0
ey	0.266	0.667	0.38
er	0.6	0.642	0.62
eh	0.538	0.714	0.613
dh	0.428	0.692	0.52
d	0.49	0.578	0.53
ch	0.0	0.0	0.0
b	0.0	0.0	0.0
ay	0.44	0.4	0.419
aw	0.33	0.428	0.37
ah	0.44	0.705	0.54
ae	0.33	0.428	0.372
aa	0.52	0.379	0.438

Table 5.43: 11F CNN-HMM - Phonemes Scores

Class	Precision	Recall	F1 Score
z	0.25	0.27	0.26
y	0.0	0.0	0.0
w	0.414	0.42	0.41
v	0.25	0.143	0.19
uw	0.166	0.125	0.143
uh	0.17	0.266	0.205
th	0.667	0.333	0.44
t	0.185	0.1667	0.175
sh	0.0	0.0	0.0
s	0.0	0.0	0.0
r	0.15	0.172	0.161
p	0.14	0.153	0.145
oy	0.103	0.15	0.122
ow	1.0	0.111	0.199
ng	0.33	0.167	0.222
n	0.281	0.236	0.26
m	0.295	0.309	0.31
l	0.0	0.0	0.0
k	0.27	0.14	0.19
jh	0.22	0.12	0.16
iy	0.12	0.15	0.133
ih	0.35	0.214	0.266
hh	0.0	0.0	0.0
g	0.36	0.381	0.372
f	0.0	0.0	0.0
ey	0.147	0.27	0.192
er	0.39	0.38	0.385
eh	0.42	0.653	0.512
dh	0.34	0.692	0.461
d	0.46	0.468	0.465
ch	0.0	0.0	0.0
b	0.0	0.0	0.0
ay	0.375	0.3	0.33
aw	0.14	0.142	0.14
ah	0.37	0.588	0.45
ae	0.2	0.29	0.235
aa	0.19	0.172	0.19

Similarly, viseme classification was also done and the results are displayed in Table 5.44 below:

Table 5.44: Viseme-wise classification using CNN-HMM

Speaker	**N_Components(States) at Max Accuracy**	**Test Accuracy(%)**
01M	7	51.09
11F	7	46.052

The confusion matrices for these models are given in Figures 5.28 and 5.29.

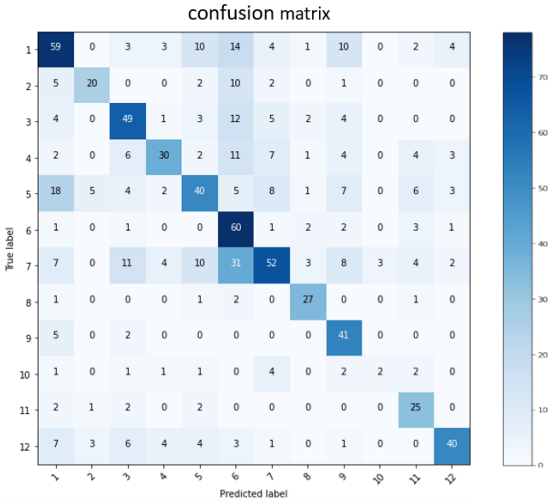

Figure 5.28: 01M CNN-HMM Viseme Confusion Matrix

Table 5.45: 01M CNN-HMM - Visemes Scores

Class	Precision	Recall	F1 Score
V1	0.49	0.503	0.495
V2	0.68	0.5	0.57
V3	0.57	0.62	0.59
V4	0.667	0.44	0.53
A	0.51	0.39	0.441
B	0.42	0.84	0.56
C	0.541	0.352	0.42
D	0.71	0.9	0.79
E	0.512	0.854	0.64
F	0.4	0.13	0.19
G	0.52	0.76	0.617
H	0.754	0.555	0.639

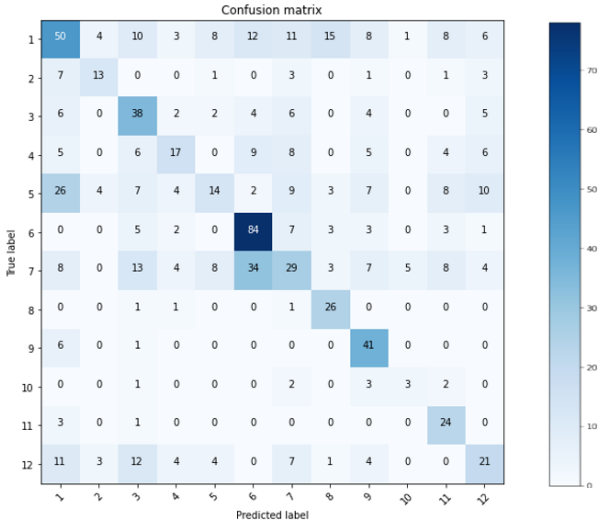

Figure 5.29: 11F CNN-HMM Viseme Confusion Matrix

Table 5.46: 11F CNN-HMM - Visemes Scores

Class	Precision	Recall	F1 Score
V1	0.41	0.368	0.387
V2	0.542	0.448	0.49
V3	0.4	0.567	0.469
V4	0.45	0.283	0.35
A	0.38	0.1489	0.21
B	0.58	0.777	0.664
C	0.35	0.236	0.281
D	0.50	0.89	0.65
E	0.49	0.85	0.626
F	0.33	0.27	0.3
G	0.413	0.85	0.558
H	0.375	0.31	0.341

A distinct difference is seen between the two models. The viseme V1 is classified well for speaker 01M but it is mis-classified many times for 11F. Also, note the difficulty in classifying the viseme F in both the matrices.

5.7 Comparison of Results

A consolidated table that compares the results of VGG16 before and after data augmentation is given below. It is clear that the PGGAN is a better architecture as it performs better than the DCGAN on all accounts. Note that 01M had 7845 images while 11F had 8710 images before augmentation. Data augmentation added 200 images to each viseme class, leading to a total of 2400 more images for each speaker.

Table 5.47: Comparison of GAN Results - Type 1 Approach

Speaker	**Before Augmentation**	**DCGAN**	**PGGAN**
01M	50.31	59.21	62.63
11F	49.42	55.48	60.52

Table 5.48: Comparison of GAN Results - Type 2 Approach

Speaker	**Before Augmentation**	**DCGAN**	**PGGAN**
01M	50.31	51.33	53.24
11F	49.42	50.00	51.61

A summary of how well all the models perform on the original images in the phoneme classes is given below:

Table 5.49: Comparison of Results across Models for Phoneme Classes

Speaker	**VGG16**	**AlexNet**	**CNN-LSTM**	**CNN-HMM**
01M	37.57	35.78	38.41	32.41
11F	35.24	24.13	36.09	30.043

A summary of how well all the models perform on the original images in the viseme classes is given below:

Table 5.50: Comparison of Results across Models for Viseme Classes

Speaker	**VGG16**	**AlexNet**	**CNN-LSTM**	**CNN-HMM**
01M	50.31	47.04	57.14	51.09
11F	49.42	45.55	53.40	46.052

Chapter 6

Conclusion and Future Work

6.1 Conclusions

This project describes and implements various models that are currently being used in the field of Visual Speech Recognition (VSR). Additionally, it incorporates the up-and-coming data augmentation technique of using GANs to generate new images that are similar to the ones in the original dataset, which can then be added to the dataset to try and increase performance. Right from the start, it was expected that viseme-wise accuracy would be much higher than phoneme-wise accuracy due to the fact that many phonemes have the same viseme (there is a many to one mapping between phonemes and visemes as seen in Neti's mapping in section 3.2) and this was as expected in the results.

The LSTM hybrid models outperform the base CNN ones on all accounts. This is understandable as the LSTM network captures temporal information, thus providing more information for proper classification. However, the HMM model surprisingly fails to perform much better than the base classification done using VGG16 even though it has the ability to capture temporal information.

Additionally, one of the two methods that incorporated data augmentation using images generated from the conditional DCGAN, namely, the type 1 method, led to a sizeable increase in testing accuracy but the other method, type 2, led to a slight increase in accuracy. A similar trend was seen when data augmentation was done using PGGAN images. This might be due to two major reasons. Firstly, the generated images did not have enough variety, and sec-

ondly, the generated images just weren't as good as the original ones, so using them for validation was not ideal. However, note that the experiments done using the PGGAN images had better results across the board compared to the ones done using the conditional DCGAN as expected, as the PGGAN is known to generate images having better quality as well as diversity.

6.2 Future Work

For further research involving the use of GANs for data augmentation in lip reading, different types of GAN architectures can be explored that can yield more diversity and quality in the generated images. Also, another interesting feature that can be included is to generate images that are related temporally, that is, to use GANs to generate a sequence of images that make up a phoneme. Another method that has scope for exploration is the use of contours as done for phoneme classification using AlexNet. However, this might not be viable for real-time recognition as making contours over each frame is infeasible. Finally, the major scope for this project is as a tool for creating word recognition algorithms in VSR that uses visemes as the fundamental unit.

Bibliography

[1] "Image captioning," https://medium.com/@sushe012/image-captioning-dec977938d0e, accessed: 2020-04-28.

[2] "Understanding LSTM networks," https://colah.github.io/posts/2015-08-Understanding-LSTMs/, accessed: 2020-04-28.

[3] "An intuitive introduction to generative adversarial networks (GANs)," An Intuitive Introduction to GANs, accessed: 2020-04-28.

[4] A. Graves, S. Fernández, and J. Schmidhuber, "Bidirectional lstm networks for improved phoneme classification and recognition," in *International Conference on Artificial Neural Networks*. Springer, 2005, pp. 799–804.

[5] D. Yu and L. Deng, *AUTOMATIC SPEECH RECOGNITION*. Springer, 2016.

[6] G. Potamianos, C. Neti, J. Luettin, and I. Matthews, "Audio-visual automatic speech recognition: An overview," *Issues in visual and audio-visual speech processing*, vol. 22, p. 23, 2004.

[7] J. Li, L. Deng, Y. Gong, and R. Haeb-Umbach, "An overview of noise-robust automatic speech recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 4, pp. 745–777, 2014.

[8] J. Huang and B. Kingsbury, "Audio-visual deep learning for noise robust speech recognition," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 7596–7599.

[9] H. Jiang and L. Deng, "A robust compensation strategy for extraneous acoustic variations in spontaneous speech recognition," *IEEE transactions on speech and audio processing*, vol. 10, no. 1, pp. 9–17, 2002.

[10] B. Shillingford, Y. Assael, M. W. Hoffman, T. Paine, C. Hughes, U. Prabhu, H. Liao, H. Sak, K. Rao, L. Bennett *et al.*, "Large-scale visual speech recognition," *arXiv preprint arXiv:1807.05162*, 2018.

[11] G. J. Wolff, K. V. Prasad, D. G. Stork, and M. Hennecke, "Lipreading by neural networks: Visual preprocessing, learning, and sensory integration," in *Advances in neural information processing systems*, 1994, pp. 1027–1034.

[12] C. Neti, G. Potamianos, J. Luettin, I. Matthews, H. Glotin, D. Vergyri, J. Sison, and A. Mashari, "Audio visual speech recognition," IDIAP, Tech. Rep., 2000.

[13] S. Dupont and J. Luettin, "Audio-visual speech modeling for continuous speech recognition," *IEEE transactions on multimedia*, vol. 2, no. 3, pp. 141–151, 2000.

[14] G. Potamianos, C. Neti, G. Gravier, A. Garg, and A. W. Senior, "Recent advances in the automatic recognition of audiovisual speech," *Proceedings of the IEEE*, vol. 91, no. 9, pp. 1306–1326, 2003.

[15] Y. LeCun, P. Haffner, L. Bottou, and Y. Bengio, "Object recognition with gradient-based learning," in *Shape, contour and grouping in computer vision*. Springer, 1999, pp. 319–345.

[16] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," California Univ San Diego La Jolla Inst for Cognitive Science, Tech. Rep., 1985.

[17] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[18] L. Rabiner and B. Juang, "An introduction to hidden markov models," *ieee assp magazine*, vol. 3, no. 1, pp. 4–16, 1986.

[19] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[20] Y. M. Assael, B. Shillingford, S. Whiteson, and N. De Freitas, "Lipnet: End-to-end sentence-level lipreading," *arXiv preprint arXiv:1611.01599*, 2016.

[21] J. R. Movellan, "Visual speech recognition with stochastic networks," in *Advances in neural information processing systems*, 1995, pp. 851–858.

[22] A. B. Hassanat, "Visual speech recognition," *Speech and Language Technologies*, vol. 1, pp. 279–303, 2011.

[23] H. Yehia, P. Rubin, and E. Vatikiotis-Bateson, "Quantitative association of vocal-tract and facial behavior," *Speech Communication*, vol. 26, no. 1-2, pp. 23–43, 1998.

[24] I. Matthews, T. F. Cootes, J. A. Bangham, S. Cox, and R. Harvey, "Extraction of visual features for lipreading," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 2, pp. 198–213, 2002.

[25] G. Zhao, M. Barnard, and M. Pietikainen, "Lipreading with local spatiotemporal descriptors," *IEEE Transactions on Multimedia*, vol. 11, no. 7, pp. 1254–1265, 2009.

[26] A. Garg, J. Noyola, and S. Bagadia, "Lip reading using cnn and lstm," Technical report, Stanford University, CS231n project report, Tech. Rep., 2016.

[27] A. Rekik, A. Ben-Hamadou, and W. Mahdi, "A new visual speech recognition approach for rgb-d cameras," in *International Conference Image Analysis and Recognition*. Springer, 2014, pp. 21–28.

[28] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[29] D. Lee, J. Lee, and K.-E. Kim, "Multi-view automatic lip-reading using neural network," in *Asian conference on computer vision*. Springer, 2016, pp. 290–302.

[30] I. Anina, Z. Zhou, G. Zhao, and M. Pietikäinen, "Ouluvs2: A multi-view audiovisual database for non-rigid mouth motion analysis," in *2015 11th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*, vol. 1. IEEE, 2015, pp. 1–5.

[31] D. Parekh, A. Gupta, S. Chhatpar, A. Yash, and M. Kulkarni, "Lip reading using convolutional auto encoders as feature extractor," in *2019 IEEE 5th International Conference for Convergence in Technology (I2CT)*. IEEE, 2019, pp. 1–6.

[32] J. S. Chung and A. Zisserman, "Lip reading in the wild," in *Asian Conference on Computer Vision*, 2016.

[33] M. Cooke, J. Barker, S. Cunningham, and X. Shao, "An audio-visual corpus for speech perception and automatic speech recognition," *The Journal of the Acoustical Society of America*, vol. 120, no. 5, pp. 2421–2424, 2006.

[34] S. Petridis, Z. Li, and M. Pantic, "End-to-end visual speech recognition with lstms," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 2592–2596.

[35] E. K. Patterson, S. Gurbuz, Z. Tufekci, and J. N. Gowdy, "Cuave: A new audio-visual database for multimodal human-computer interface research," in *2002 IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 2. IEEE, 2002, pp. II–2017.

[36] G. W. Taylor, R. Fergus, Y. LeCun, and C. Bregler, "Convolutional learning of spatio-temporal features," in *European conference on computer vision*. Springer, 2010, pp. 140–153.

[37] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[38] K. Noda, Y. Yamaguchi, K. Nakadai, H. G. Okuno, and T. Ogata, "Lipreading using convolutional neural network," in *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[39] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[40] E. Tatulli and T. Hueber, "Feature extraction using multimodal convolutional neural networks for visual speech recognition," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 2971–2975.

[41] D. Wu and Q. Ruan, "Lip reading based on cascade feature extraction and hmm," in *2014 12th International Conference on Signal Processing (ICSP)*. IEEE, 2014, pp. 1306–1310.

[42] P. Sujatha and M. R. Krishnan, "Lip feature extraction for visual speech recognition using hidden markov model," in *2012 International Conference on Computing, Communication and Applications*. IEEE, 2012, pp. 1–5.

[43] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[44] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[45] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[46] E. Wu, K. Wu, D. Cox, and W. Lotter, "Conditional infilling gans for data augmentation in mammogram classification," in *Image Analysis for Moving Organ, Breast, and Thoracic Images*. Springer, 2018, pp. 98–106.

[47] M. Frid-Adar, E. Klang, M. Amitai, J. Goldberger, and H. Greenspan, "Gan-based data augmentation for improved liver lesion classification," 2018.

[48] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[49] M. J. Chuquicusma, S. Hussein, J. Burt, and U. Bagci, "How to fool radiologists with generative adversarial networks? a visual turing test for lung cancer diagnosis," in *2018 IEEE 15th international symposium on biomedical imaging (ISBI 2018)*. IEEE, 2018, pp. 240–244.

[50] D. A. B. Oliveira, A. B. Mattos, and E. da Silva Morais, "Improving viseme recognition using gan-based frontal view mapping," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. IEEE, 2018, pp. 2229–22 297.

[51] N. Harte and E. Gillen, "Tcd-timit: An audio-visual corpus of continuous speech," *IEEE Transactions on Multimedia*, vol. 17, no. 5, pp. 603–615, 2015.

[52] G. Bradski, "The opencv library," *Dr Dobb's J. Software Tools*, vol. 25, pp. 120–125, 2000.

[53] "Facial mapping (landmarks) with dlib + python," Face Detection with dlib, accessed: 2020-04-28.

[54] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *CVPR09*, 2009.

[55] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 807–814.

[56] D. A. Shoieb, S. M. Youssef, and W. M. Aly, "Computer-aided model for skin diagnosis using deep learning," *Journal of Image and Graphics*, vol. 4, no. 2, pp. 122–129, 2016.

[57] D. Jurafsky, *Speech & language processing*. Pearson Education India, 2000.

[58] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," in *Readings in speech recognition*, 1990, pp. 267–296.

[59] L. Rabiner and B. Juang, "An introduction to hidden markov models," *ieee assp magazine*, vol. 3, no. 1, pp. 4–16, 1986.

[60] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014.

[61] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. icml*, vol. 30, no. 1, 2013, p. 3.

[62] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of gans for improved quality, stability, and variation," *arXiv preprint arXiv:1710.10196*, 2017.

[63] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.