Lip reading using 3D Convolutional Neural Networks

Aditya Rangarajan , Avinash Kumar, Megh Bhalerao and B Vrushabh

April 20, 2019

***Abstract*— This report aims to present a methodology for Lip reading using 3D convolutional neural networks. Convolutional neural networks are a state of the art method. VidTIMIT corpus is a data set rich in audio visual data, which is used in this work.In the proposed method, we aim to separately pre-process the audio and the visual data. From the distinguished visual data, the Lip Region and its variations, as each word is spoken is extracted for further processing. Histogram of gradients is used for extraction of the facial features. These gradients can be used to describe the overall appearance of any image. From the Audio Data, we extract the speech content and denoise,enhance and further process it to reduce the noise and capture the intricacies in speech variation in the due course of lip movement.**

I. INTRODUCTION

Detection and recognition of lip movements has been a very active field of human research. Detection of the lip movements not only provide an insight into Voice Activity Detection(VAD) but also aids in fields like gesture recognition and emotion detection. This work maps the 'voice-activity' with the 'lip-activity' and trains a 3D convolutional network to quantify the relation between the two. There are numerous approaches in the past which have successfully applied deep learning to VAD such as the work of Tamura et al[1] and Lee et al[2]. In the work of Tamura et al[1], a tandem approach[3] is used where deep learning is applied to estimate probabilities on the Hidden Markov Model (HMM) states for the test data. While Lee et al[2] attempts to correlate image processing with audio processing, where extraction of the lip movement signal with the speech signal is done and further compared for verification and processing.

In our work we attempt to understand and implement voice activity detection with a 3D convolutional network, taking into account variable factors like noisy audio or changing lighting conditions. The motivation for implementing such a system arises from the work of Torfi et al[4]. In the work of Torfi et al[4], the speech and the visual aspect of training for the network are impemented separately and then fed to the 3D convolutional network for prediction and generalization. In our work, we also follow such a method of processing and implementation.In the work of Wentao et al[5],MMSE-STSA based estimation is stated, which incorporates the estimation of spectral component of the speech signal.

II. DATASET

The VidTIMIT corpus is a dataset rich in audio visual data. This dataset consists of 43 test subjects. Each subject is made to speak 10 sentences chosen out of the TIMIT corpus. A broadcast quality video camera is used for recording. The video has a resolution of 512 x 384 pixels.The data pertaining to each session is stored in a mono, 16 bit, 32 KHz WAV file.In every session corresponding to each subject, a head rotation has been performed.Preprocessing is performed on this dataset before incorporating it into our work.The video frames do not require any preprocessing before incorporation in our work, but the audio corresponding to the videos needs to be denoised, this is further discussed in section IIIB.

III. PROPOSED METHODOLOGY

A. Visual Enhancement

The video frames corresponding to each test subject is subjected to facial feature detection. This is implemented using the **dlib** library in python. The library function $get\_frontal\_face\_detector()$ is used. This library utilizes **Histogram of Oriented Gradients** (HOG) along with a **linear classifier** for feature extraction in the facial region.The HOG features are obtained by dividing the image into smaller subsections and plotting the histogram for the magnitude and angle. The principle behind using HOG is that the appearance of any image is composed of a distribution of intensity gradients or the directions of the edges.The image is divided into small connected regions called cells, and for the pixels within each cell,a HOG direction is compiled. A concatenation of these directions ultimately aid to find the facial features.The HOG features locate 68 points in the human facial region which includes areas like the face outline, nose and the mouth region. The indices of the region of interest i.e., the lip region begins from the 48^{th} point and ends at the 68^{th} point. Using the coordinates of these sets of points obtained as an output to the $get\_frontal\_face\_detector()$ function, the mouth region is extracted from each frame. These video frames contain noise relating to the head motion while the subject is speaking. To eliminate the effect of the same, a cropping window has to be designed to take care of such extremities. The cropping window must be able to divide the lip region based on a specified threshold to account the variations in head motion.This threshold is decided dynamically based on whether the collection of points corresponding to the lip region fall inside the cropping window or outside it.This method of feature extraction is tested in different conditions of light and varying head motions, an illustration of the same is shown in Fig.1 . The cropped images of the lip region(Fig. 3) are made into a cube i.e., a collection of these images are fed to the convolutional network as one of its inputs. A brief overview of the preprocessing is shown in Fig.2.

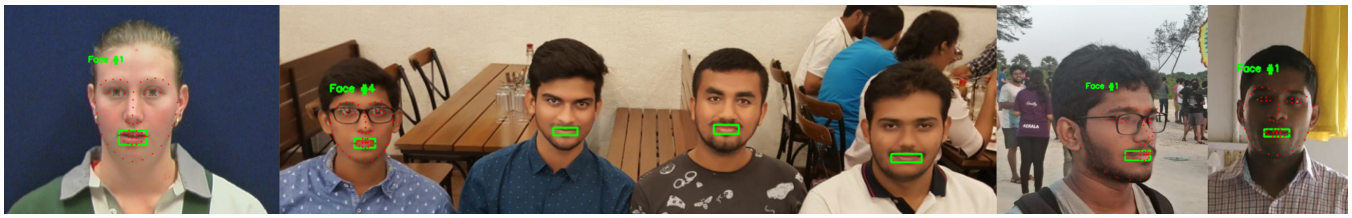

Fig. 1: Facial feature detection and lip isolation in (From left to right) (a) Normal conditions (b) A group of people (c) A different head orientation (d) Different lighting conditions

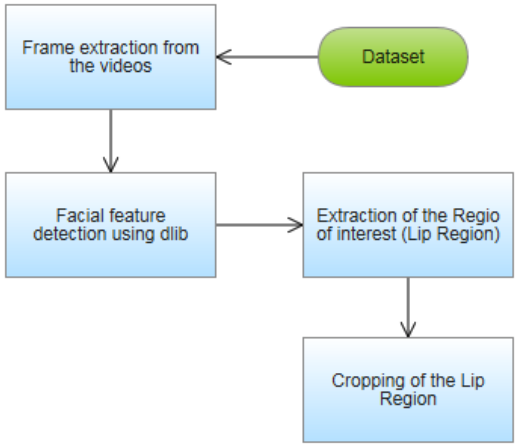

Fig. 2: Flowchart of Visual preprocessing

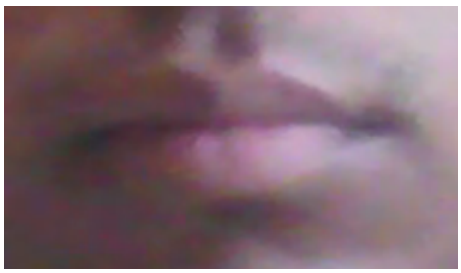

Fig. 3: Image of a Cropped Lip

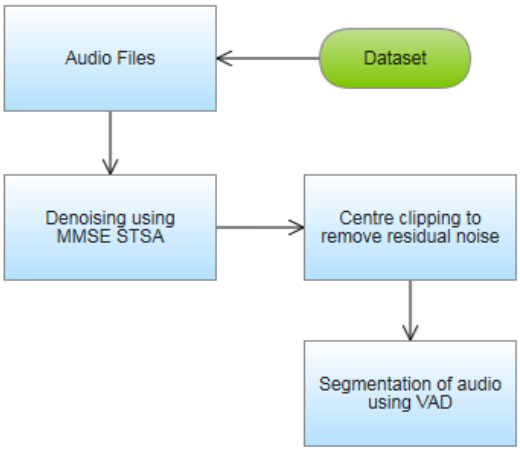

Fig. 4: Flowchart of Audio preprocessing

B. Speech Enhancement

We employed the algorithm mentioned in [5]. We incorporated the noisy speech Enhancement algorithms based on Minimum Mean-Square-Error Short time Spectral speech production and perception. They fall into three classes:
(1)Enhancement based on Short-time Spectral Amplitude estimation.
(2)Enhancement based on Periodicity of voiced speech.
(3)Enhancement based on a speech model.

Residual noise always exists after processing. To reduce it, we apply center clipping to the processed speech, and modify the gain function.

MMSE-STSA

STSA of noisy speech is calculated on each frame by FFT, with phases extracted stored, then STSA estimation of the supposed clean speech is done on every frequency modulus. The speech waveform is reconstructed from the estimated spectral amplitude and the stored phase. Let x(t) and d(t) denote the speech and the noise, respectively. The noisy speech y(t) is given by

y(t) = x(t) + d(t) , $0 < t < T(1)$

Let X_k, = $A_k$exp(jα_k),D_k, and Y_k = $R_k$exp(jθ_k) denote the k^{th} spectral component of the signal x(t), the noise d(t), and the noisy signal y(t), respectively. Our task is to estimate x(t) from y(t), or to estimate x , from $\{Y_0, Y_1,...\}$. Since we are interested in only spectral amplitude ,the estimation problem can be simplified to estimating A from $\{Y_0, Y_1,...\}$. In addition, based om the statistical independence assumption ,of the spectral components, the MMSE estimator $\bar{A}_k$ of A_k is derived as follows:

$$\bar{A}_k = E\{A_k \ |Y_0, Y_1...\} = E\{A_k|Y_k\}$$

$$= \frac{\int_0^{+\infty}\int_0^{2\pi} a_k P(Y_k|P(Y_k|a_k,\alpha_k))P(q_k,a_k)\,da_k\,d\alpha_k}{\int_0^{+\infty}\int_0^{2\pi} a_k P(Y_k|a_k,\alpha_k)P(\alpha_k,a_k)\,da_k\,d\alpha_k} (1)$$

where $P(Y_k|a_k,\alpha_k)$ and $P(a_k,\alpha_k)$ are given by

$$P(Y_k|a_k,\alpha_k) = \frac{1}{\pi\lambda_d(k)} exp \frac{-1}{\lambda_d(k)(|Y_k - a_k \exp j\alpha_k|)^2} (2)$$

$$P(a_k,\alpha_k) = \frac{a_k}{\pi\lambda_x(k)} exp \frac{-{\alpha_k}^2}{\lambda_x(k} (3)$$

in which $\lambda_x(k) = E\{|X_k^2|\}$ and $\lambda_d(k) = E\{|D_k^2|\}$ are the variance in the k^th spectral component of the speech and

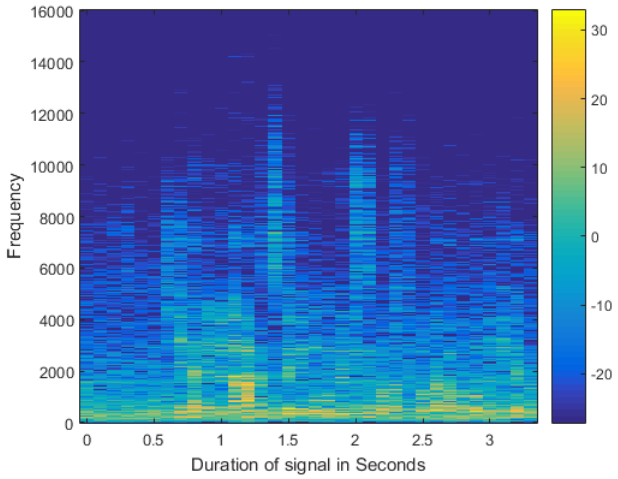

Fig. 5: Spectrogram of Noisy Speech

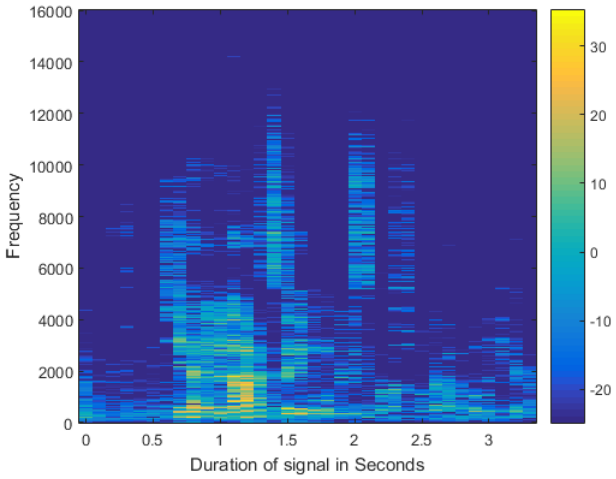

Fig. 6: Spectrogram of denoised Speech

noise respectively. Substituting (3) and (4) into (2) ,we get

$$\bar{A}_k = \tau(1.5)\frac{\sqrt{\nu_k}}{\gamma_k} M(-0.5; 1; -\gamma_k) R_k \tag{4}$$

$\tau(.)$ denotes the gamma function, with $\tau(1.5) = \sqrt{pi}/2$; ν_k is defined by,

$$\nu_k = \frac{\epsilon_k}{1+\epsilon_k}\gamma_k \tag{5}$$

where ϵ_k and γ_k are defined by

$$\epsilon_k = \frac{\lambda_x(k)}{\lambda_d(k)} \tag{6}$$

$$\nu_k = \frac{{R_k}^2}{{}_d(k)} \tag{7}$$

ϵ_k and ν_k are interpreted as the a priori and a posteriori SNR, respectively. Fig 4 gives a brief flow diagram of the audio preprocessing.

In the mentioned figures we compare the spectrograms of noisy speech(Fig 5),speech post the action of MMSE-STSA algorithm(Fig 6) and time scale comparison of Noisy Speech and Enhanced Speech (Fig7).

C. Segmentation using VAD

The basic principle of VAD is to extract measured features and compare it against a set threshold. If the measured values are greater than the threshold, voice activity is detected and VAD is set to 1. Otherwise, VAD is set to zero and silence is present in that interval. Most VAD algorithms output a binary decision for each frame. The accuracy and reliability of a VAD algorithm depend heavily on the decision thresholds.

VAD Algorithms based on energy thresholding[6] :
In energy based VAD, speech is detected if energy is greater than the threshold.

$$IF(E_j > k.E_r), where\ k > 1, frame\ is\ ACTIVE$$

$$ELSE\ frame\ is\ INACTIVE$$

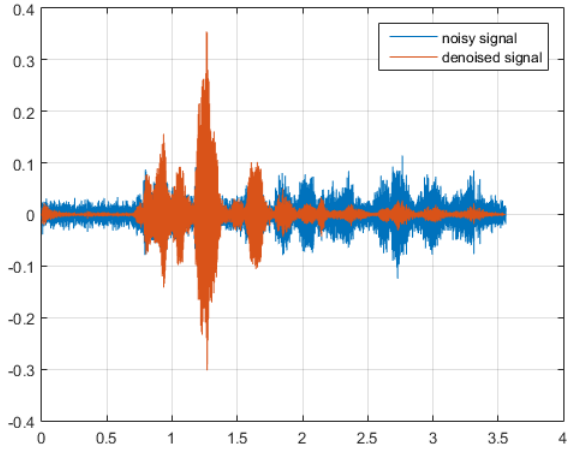

Fig. 7: Comparison of Noisy Speech and Enhanced Speech

Where Er is the energy of the noise frames and K.Er is the used threshold. The scaling factor allows a buffer to adapt Er and hence the threshold. The rule for updating the threshold is:

$$E_{rnew} = (1-p).E_{rold} + p.E_{silence} \tag{8}$$

Where E_{rnew} is the updated value of the threshold, E_{rold} is the previous energy threshold and E_{silence} is the most recent undetected frame. The parameter P is a constant(0¡p¡1).

Energy of a frame There are two common ways to estimate the energy of a frame -

1) Short-term energy
2) Root Mean square energy(RMSE)

If x(i) is the i^th sample of the speech, N is the number of samples in a frame, then the short term energy of the j^th frame of a speech signal can be represented as:

$$E_j = \frac{1}{N}. \sum_{i=(j-1).N+1}^{j.N} x(i)^2 \tag{9}$$

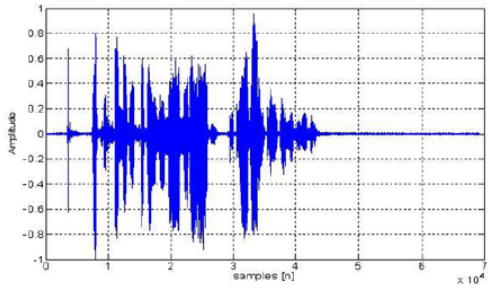

Fig. 8: Input Audio Signal

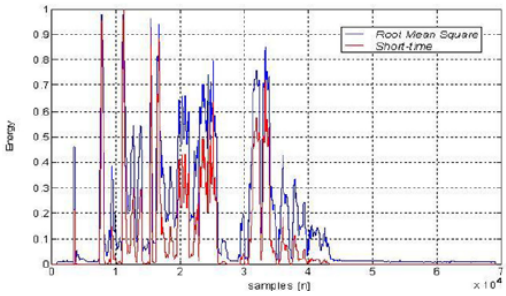

Fig. 9: Short time vs Root Mean square Energy

And the root mean square energy can be represented as

$$E_j = [\frac{1}{N}. \sum_{i=(j-1).N+1}^{j.N} x(i)^2]^{1/2} \tag{10}$$

The energy estimate of speech exhibits peaks and valleys. The peaks correspond to activity while the valleys can be used to obtain a noise power estimate. The RMSE is more appropriate for thresholding, as it displays valleys in greater detail.

Feature Extraction

The feature extraction is done for a periodicity measure of the signal by determining the estimated pitch period of the signal, by first center clipping followed by normalized autocorrelation function.

$$R(\tau) = \frac{\sum_{n=0}^{N-m-1} x(n).x(n+r)}{\sqrt{\sum_{n=0}^{N-m-1} x^2(n+r)}} \tag{11}$$

$$T_{min} \leq \tau \leq T_{max} \tag{12}$$

The pitch period of a voiced frame is equal to the value of that maximizes the normalized autocorrelation function. The total voice band energy Ef is computed for the voice band frequency range from 0 Hz to 4 kHz, by using the regular signal energy equation. The threshold for computing the voiceband is calculated using :

$$Threshold = (1 - \lambda)E_{max} + \lambda E_{min}$$

Where Emax and Emin are the voiceband energy levels obtained from incoming frames and λ is an adaptive number different for different types of signals. After feature extraction, the parameters are compared with several thresholds to generate an initial VAD decision. The detailed flow of generation of a final VAD decision is given beside in the form of a flowchart (Fig 11)

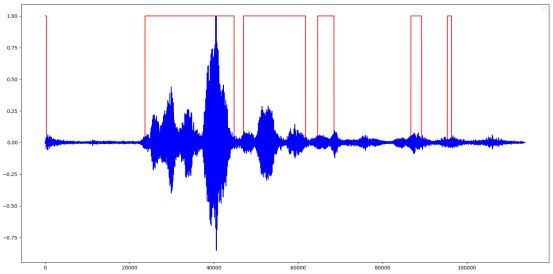

Fig. 10: Voice Activity Detected segments

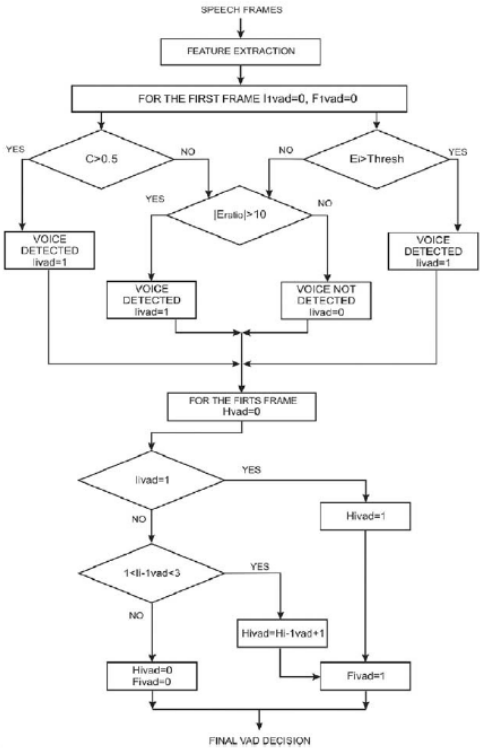

Fig. 5: Detailed flowchart of the proposed VAD

Fig. 11: Flowchart of Voice Activity Detection (VAD)

1) Adaptive VAD: Using the above method to segment words is a very inefficient one as we have to manually set the threshold for what is silence and what is not. To do this manually for all the audio files would be a painstakingly ling process. Also to decide upon a correct threshold to segment each word takes a lot of trial and error. So to arrive at a threshold and applying the threshold for all the audio files in the dataset would take a lot of time. So we used an adaptive thresholding which does this job for us. This algorithm assumes that the first few frames of the audio files are silence and based on that it will identify the

threshold to be applied for the entire audio file. The problem with using VAD that it is almost impossible to get each word in separate frames. This is because there isn't a well defined period of silence between the words. So using VAD wee can get segments that contain multiple words, that is the sentence is segmented into phrases. The adaptive VAD allows us to achieve this thing. Using the simple VAD it is very difficult to even segment the sentences into legilbe phrases, most of the times the sentence gets segmented into segments that are random sounds.

D. Preprocessing the Data using FFmpeg

FFmpeg is a free software project consisting of a vast number of libraries and packages that allow you to handle audio, video and other multimedia files effectively. The FFmpeg software was used to preprocess the files of the LRW dataset. The following steps were employed:

- First, the portion of the video which contains the word was extracted.
- Then, its frame rate was changed from 25 fps to 30 fps.
- Then 9 frames of the video were selected such that the duration of each word is around 0.3 seconds.
- After this the image and the audio part were separated and stored in separate files.

These steps are to applied to each video file in the LRW dataset, but due to the immense size of the dataset and lack of computational resources we could not try the preprocessing on the entire dataset. After this initial set of preprocessing ins done, further preprocessing is done to denoise the audio file as explained earlier in the report. After all the preprocessing is completed, the required features are extracted from the video and the audio files. The extracted features are then fed to the neural network for training. The FFmpeg commands used to achieve the above task are depicted in Figure 12

```
#To choose a particular segment of the video
ffmpeg -i source.mp4 -ss 00:00:05 -t 00:00:10 -c copy cut_video.mp4


#Changing the frame rate
ffmpeg -y -i seeing_noaudio.mp4 -vf "setpts=1.25*PTS" -r 24 seeing.mp4


#To copy only the video(withput audio)
ffmpeg -i ABOUT_00001.mp4 -an -c copy video.mp4


#To store only the audio
ffmpeg -i test.mp4 -ab 160k -ac 2 -ar 44100 -vn audio.wav
```

Fig. 12: FFmpeg commands used

IV. CONVOLUTIONAL NEURAL NETWORK

Subsequent to feature extraction of the Audio and Video Datasets the next task is to train a CNN on these extracted features ($500 \times 1000 = 500,000$ features), where 500 is the number of words and 1000 is the number of instances of occurrence of the word.

The Proposed is a 3D CNN architecture since the features themselves are 3 Dimensional - the video is a temporal feature cube while the audio is a 3D Spectrogram feature cube. The architecture is composed of two coupled CNN with a common loss function - one network for the audio and the other for the video. The two networks are coupled at their highest level i.e. the last fully connected layer. A contrastive loss function of the following form is used :

$$L_W(X,Y) = \frac{1}{N}\sum_{i=1}^{N} L_W(Y_i,(X_{p_1},X_{p_2})_i)$$

The individual terms are explained as follows :

1) X_{p_1}, X_{p_2} are the i^{th} training sample pair of the visual and audio CNN.
2) Y_i is the ground truth of the i^{th} training sample

L_W is defined as the following :

$$L_W(Y_i,(X_{p_1},X_{p_2})_i) =$$
$$Y \times L_{gen}(D_W(X_{p_1},X_{p_2})_i)$$
$$+(1-Y)\times L_{imp}(D_W(X_{p_1},X_{p_2})_i) + \lambda||W||_2$$

Here the regularization parameter λ is used for over-fitting prevention. D_W is the Euclidean distance between the input datapoints. L_{gen} and L_{imp} are known as the genuine and imposter losses respectively.
Both the neural networks have a dropout layer, again to prevent overfitting, There is no such method so as to choose the "right" network architecture. We try time and again with newer architectures and observe if there is any accuracy improvement. If there is an accuracy improvement on the testing dataset then we choose that architecture over the previous one. The specifications of the visual and audio CNN are given below in a tabular format respectively :

Layer	Input-Size	output - size	kernel	stride
Conv1	(90,60,100,1)	(7,58,98,16)	(3,3,3)	1
Pool 1	(7,58,98,16)	(7,28,48,16)	(1,3,3)	(1,2,2)
Conv 2	(7,28,48,16)	(5,26,46,32)	(3,3,3)	1
Pool 2	(5,26,46,32)	(5,12,22,32)	(1,3,3)	(1,2,2)
Conv 3	(5,12,22,32)	(3,10,20,64)	(3,3,3)	1
Pool 3	(3,10,20,64)	(3,4,9,64)	(1,3,3)	
Conv 4	(3,4,9,64)	(1,2,7,128)	(3,3,3)	
FC 5	(1,2,7,128)	256		
FC 6	256	64		

TABLE I: Architecture of Visual CNN

Layer	Input-Size	output - size	kernel	stride
Conv1	(15,40,3)	(7,58,98,16)	(3,3,3)	1
Pool 1	(13,36,1,16)	(7,28,48,16)	(1,3,3)	(1,2,2)
Conv 2-1	(13,18,1,16)	(5,26,46,32)	(3,3,3)	1
Conv 2-2	(11,15,1,32)	(9,12,1,32)	3,4,1	1
Pool 2	(9,12,1,32)	(9,6,1,32)	(1,2,1)	(1,2,1)
Conv 3-1	(9,6,1,32)	(7,4,1,64)	(3,3,1)	1
Conv 3-2	(7,4,1,64)	(5,2,1,64)	(3,3,1)	1
Conv 4	(5,2,1,64)	(3,1,1,128)	(3,2,1)	1
FC 5	(3,1,1,128)	64	-	-

TABLE II: Architecture of Audio CNN

Here we can make an observation that every output is a 4 dimensional tensor since our input itself is 3 Dimensional and hence by convolving with multiple kernels the 3D inputs are converted in 4D tensors. The Pooling layers use a function called maxpool where the maximum of a given number of pixels is taken and that forms your output which is fed to the next layer.

V. PROBLEMS FACED AND COMPUTATIONAL COST

1) The LRW dataset used here is an immensely large dataset consequently increasing the computational cost.
2) Neural Network Training and Feature extraction would require a great deal of computational power and a large number of GPU hours, not available to us presently.
3) Google Colaboratory provides free GPU access for up to 12 hours a day which could be used in the CNN training, but the problem here is that the dataset cannot be accessed locally from the system but has to be uploaded on Google drive first before accessing.
4) But, such a large 100 GB dataset cannot be uploaded on Google Drive.
5) Since, the required computational resources are not available we tried to train the neural network with a random dataset of the same dimensions of the extracted feature vectors, since the feature vectors are nothing but matrices of real numbers.
6) We are still in the process of debugging the Network.
7) Once, the network gets trained on the random dataset, and assuming we obtain enough computational resources we can train and test the network on the actual LRW dataset and evaluate the network accuracy.

VI. CONCLUSIONS

In our work we have demonstrated the preprocessing of the audio and video sections of the VidTIMIT & the Lip Reading in the Wild (LRW - by BBC) audio visual data which will be later fed to the 3D Convolutional Neural Network for training. The convolutional neural network training phase is underway, and efforts are being made to train the CNN on a random dataset first and then, if provided access with enough computational resources, we can preprocess the entire LRW Dataset and feed our neural network with these processed feature vectors. Only when the CNN is trained on the actual dataset, can we evaluate its performance. This work is not eligible presently to comment upon the accuracy of lip reading and related parameters.

VII. APPENDIX

Please find the relevant codes at : `https://github.com/meghbhalerao/Lip-Reading`

REFERENCES

[1] 'Audio-visual speech recognition using deep bottleneck features and high-performance lipreading',Satoshi Tamura,Hiroshi Ninomiya, Norihide Kitaoka, Shin Osuga, Yurie Iribe, Kazuya Takeda and Satoru Hayamizu.

[2] 'Speech Activity Detection with Lip Movement Image Signals',Soo-jong Lee, Jun Park and Eung-kyeu Kim.

[3] Yuan Yuan, Chunlin Tian, Xiaoqiang Lu, "Auxiliary Loss Multimodal GRU Model in Audio-Visual Speech Recognition", Access IEEE, vol. 6, pp. 5573-5583, 2018

[4] '3D Convolutional Neural Networks for Cross Audio-Visual Matching Recognition', Amirsina Torfi, Seyed Mehdi Iranmanesh, Nasser Nasrabadi and Jeremy Dawson .

[5] 'Speech Estimation based on MMSE-STSA Estimation and Residual noise redution',Zheng Wentao and Cao Zhigang

[6] 'Voice Activity Detection for Speech Enhancement Applications' E. Verteletskaya, K. Sakhnov