Indian Institute of Technology Bombay

Rnd Project

Audio Visual Speech Recognition

Authors:
Himanshu Pandotra
13D070037
Kanhaiya Kumar
130110008

Supervisor:
Prof. Rajbabu Velmurugan

May 2, 2017

Abstract

Speech recognition is an important problem which has been studied widely in recent years. Many approaches have been utilized for automatic speech recognition (ASR) and good results have been obtained in application specific environments.In this work,we look at the problem of Audio Visual Speech Recognition (AVSR) where we try to improve speech recognition accuracy by utilizing the visual information of lip movement of the speaker.We use a very recent deep learning based approach for AVSR by combining information in audio and visual streams and learning models through them.

1 Introduction:

1.1 Background and Motivation:

Automatic Speech Recognition has been a topic of research for several years as it has several useful applications like In-Car systems, Robotics , Voice Control Systems, transcription of Speech to text etc. Traditionally, Hidden Markov Model (HMM) based systems have been very popular for ASR. HMM based models have flexibility of working at phoneme, word or sentence level depending upon how these models are build and trained for specific applications.These techniques have been widely studied and various language models have been build for creating state of art ASR systems.

Most speech recognition systems perform very well in constrained environments where specific task information can be taken into account while building ASR systems. 'ASR in the wild' is still a hot topic of research with top firms competing to get best results.

Recently,various deep learning based method have been used for ASR. State of the art methods involve carefully combining deep learning techniques with HMM based models to get good accuracies.

Many research groups in top universities and Technical firms including Microsoft, Facebook, Google, BAIDU have shown significant interest in using machine learning techniques to keep on improving speech recognition results and achieve better accuracies.In the past 4-5 years,deep learning concepts like RNN, LSTM have become extremely popular for time-series prediction.The

Connectionist Temporal Classification (CTC) objective introduced by Alex Graves in [3] in 2014 has made possible to build end-to-end ASR systems purely based on deep learning.

Audio-visual automatic speech recognition (AV-ASR) is a multi-modal analysis in which audio and visual features are combined for speech recognition. The idea is that by incorporating visual feature information based on the speaker's lip movements into automatic speech recognition (ASR) systems,the accuracies can be significantly improved especially under noisy conditions.

1.2 Project Goal:

Here, we will evaluate an audio-visual ASR (AV-ASR) system using deep recurrent neural network (RNN) and CTC objective function.The model we build here will combine audio MFCC features and video features and then train a network to learn complex features for speech recognition and will directly output the text corresponding to the spoken words. We will test our model on the Grid Corpus dataset.[6]

2 Theory

In our model,we use various neural network components which have there own functionality and usage. Convolution layers (CNN),Recurrent Neural Network RNN(LSTM), Connectionist Temporal Classification (CTC) loss are the important parts of this network. We will look at the different components of the deep learning architecture and try to understand its usage in our system.

2.1 Convolutional Neural Network

The CNN are very similar to standard Neural Network but instead of fully connected neurons between two layers,we have only local connections.These weights act as filters which are locally connected and are moved over are the neurons in the previous layer.This is equivalent to the convolution operation with weight filters. The benefit of convolution layer is the reduced number of weight parameters to learn as they are shared across neurons because of the convolution operation. This kind of operation is very similar to our standard

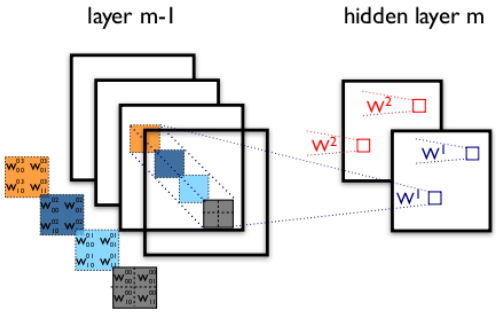


Figure 1: Convolution Layer : Source [9]

signal processing technique of convolving the signal with filters.Effectively CNN layers are used to extract patch level information from the input neurons.Typically this layer is applied at the beginning to the input images which are convolved with these filters and useful features are extracted from the images which are then fed to the next layer.

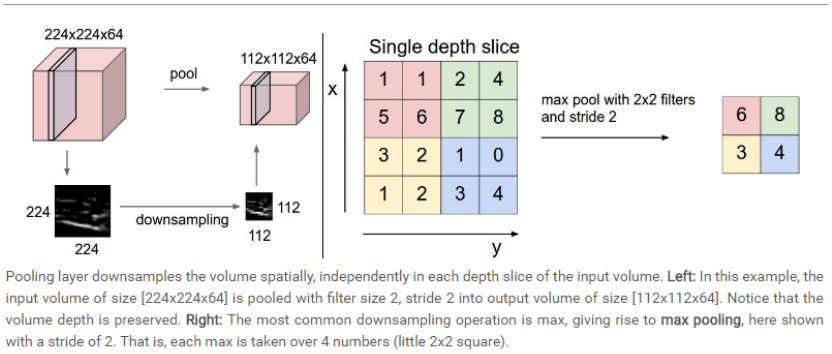


Figure 2: Max Pooling Layer : Source [8]

Max Pooling Layer is a special type of convolution layer where the filter sub samples the input at a given rate. This means for a $k \times k$ input, the output of the layer is $k/m \times k/m$, where m is the sub sampling ratio.This type of layer is utilized to prevent major growth of neurons across layers and get a smaller layer before termination or fully connected layer.The intuition behind this layer is that there is redundant information in close by neurons and we

can sub sample the to remove this redundancy allowing faster learning. This layer significantly improves performance.

2.2 Bidirectional RNN-LSTM

Recurrent Neural Network is a model which takes into account the temporal aspects of data.The neuron in RNN has a looping structure and the information from current time step is carried to the next time step. RNN are useful when current output of the network should depend on the data seen till now, for example in text prediction. This structure became popular because of its use in handling unsegmented data like speech and handwriting recognition.

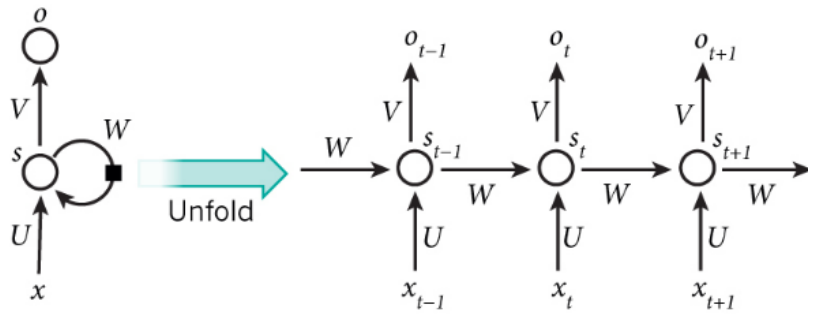


Figure 3: A neuron unfolding in time. RNN : Source [Nature]

One of the limitation's of RNN was that during the learning phase,the gradients at current time step would propagate back in time and the values become smaller and smaller at each step.This problem is known as the vanishing gradients problem.To overcome this, a variant of RNN, **Long Short Term Memory** (LSTM) cell was proposed. The LSTM cell has a gated structure which decides what information should flow. It typically consists of a memory cell/state, a forget gate, a input gate and a output gate. The forget gate decides whether information from previous time step should flow to the current state. The input gate decides, how much of current input should influence the current state. The output gate decides whether to give an output at current time step or not. So, during gradient descent via back propagation through time,the information of only few previous states is taken

into account and rest is ignored via forget gate, preventing vanishing gradients.

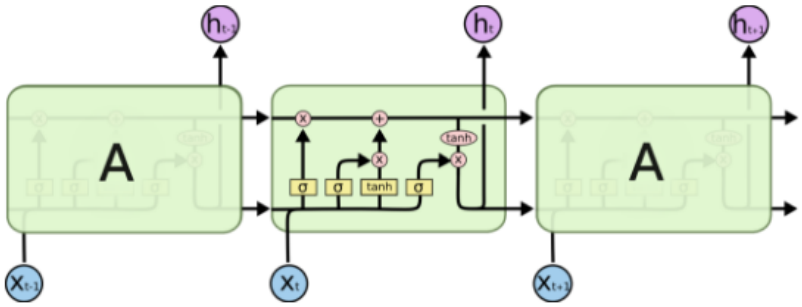


Figure 4: LSTM : Source [10]

2.3 Connectionist Temporal Classification

Connectionist Temporal Classification (CTC) first introduced here [3] is a NN layer which minimize the loss between its input sequence and the label output sequence. The use of RNN in speech recognition is made more efficient via CTC because a RNN requires pre segmented data sequence as input and gives a sequence with same number of time steps as its output.Therefore post processing is required to convert this output sequence to a label sequence.This would limit its performance.But CTC loss helps in training RNN with unsegmented sequences directly.Hence the output of the network can be directly unsegmented character sequences or words.

To explain CTC objective and sequence labelling as done by author of [1], consider a input sequence $I = (I_1, I_2, ...I_T)$ and label sequence be $l = (l_1, l_2, ...l_S)$ where $S < T$. The labels are taken from a set $|L^{'}| = |L| + 1$ where $|L|$ is the set of actual labels and one extra label ϕ is added as a blank label. The RNN outputs a sequence of length t and each of the time steps has a $|L|^{'}$ length vector containing the probabilities of having one of the $L \cup \phi$ label at that time step. The additional ϕ label helps in getting different alignments for the same output label. In our case, the set of labels are the 26 alphabets a-z and a blank label ϕ. So for example, both the alignment sequences $b\phi\phi l\phi ue$ and $bl\phi\phi\phi ue$ correspond to the same label sequence $'blue'$. There are many other sequences which correspond to same label.In fact, we

can define a many to one mapping $H : L^T -> L^S$ from alignment sequence on time steps to the actual underlying label sequence or in our case the word. Let σ represent one such alignment of length T.The posterior probability of having this alignment given the observed sequence of vectors $I = (I_1, I_2, ...I_T)$ for T time steps (where each I_i is a vector of $|L|^{'}$ length containing RNN outputs passed through softmax layer) is,

$$P(\sigma|\mathbf{I}) = \prod_{t=1}^{T} P(l_t|I_t) \tag{1}$$

where $l_t \in L^{'}$ and $\mathrm{P}(l_t|I_t)$ is given by

$$P(l_t|I_t) = \frac{exp(y_t^l)}{\sum_{l'} exp(y_t^{l'})} \tag{2}$$

where y_t^l is the input to the lth node of soft-max layer at time t. The CTC function tries to maximize the following probability.

$$P(\mathbf{label}|\mathbf{I}) = \sum_{\sigma \in H^{-1}(label)} P(\sigma|\mathbf{I}) \tag{3}$$

where σ is a possible alignment sequence for given label. **I** is the observed sequence of probability vectors.

3 Model

3.1 Feature Extraction

Audio MFCC

For audio part, we use the standard MFCC(Mel Frequency Cepstral Coefficients) features. The audio dataset is available at 50KHz rate.We apply a window frame of 10ms with a hop of 5ms and extracted the 40 dimensional MFCC vectors with cutoff at 8 KHz.(The higher frequencies generally do not contain much information) for each frame. Along with the MFCC vectors we also concatenate the delta cepstral coefficients and double delta cepstral coefficients which are basically the time derivatives and double derivatives of MFCC features.These features help in taking temporal information into account.The length of each feature vector is 120x1.

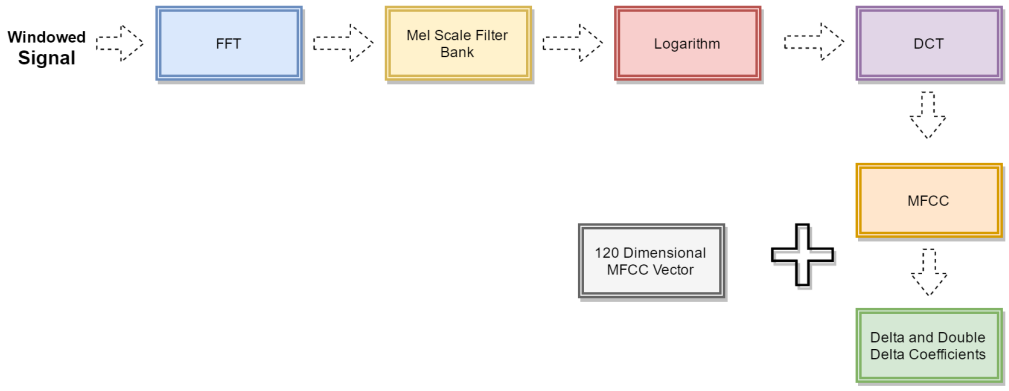
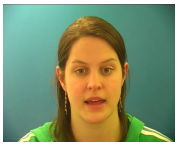


Figure 5: Audio Feature Extraction

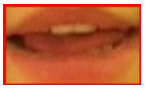
Visual Features

The relevant video features correspond to the lip portion of the person speaking.We want to identify the person speaking in the scene and from there we extract the lip region.In the general case,we may have multiple people in the scene where we need to identify face each person speaking and then separate out the audio for each person.This is a difficult problem to handle. In our case we look at a simple model based on the dataset that we are using citedata where we have a single person speaking nearly facing the camera. So here we just identify the persons face and the mouth region as shown below.For extraction of facial features,we use pre-trained facial feature detector from [11] and use opencv and dlib python libraries to separate out the mouth region in each frame of the video. This is done for each person in the dataset. The lip region frame extracted is rescaled to a standard size of 64x64. We use this 64x64 vector from each video frame as a video feature. The following is an example of extracted features:



(a) Face Image

(b) Mouth Detected



(c) lip region

Figure 6: Video Feature Detection

3.2 Architecture

After extraction of the relevant features,we need to train our models on the training data. Here, we look at three modalities for Speech Recognition.First, we use only the audio features extracted to build a speech recognition network.Then we build a network which is purely based on Visual speech recognition where only video features will be used.Finally, we try to combine these networks and build a joint audio+visual speech recognition system.

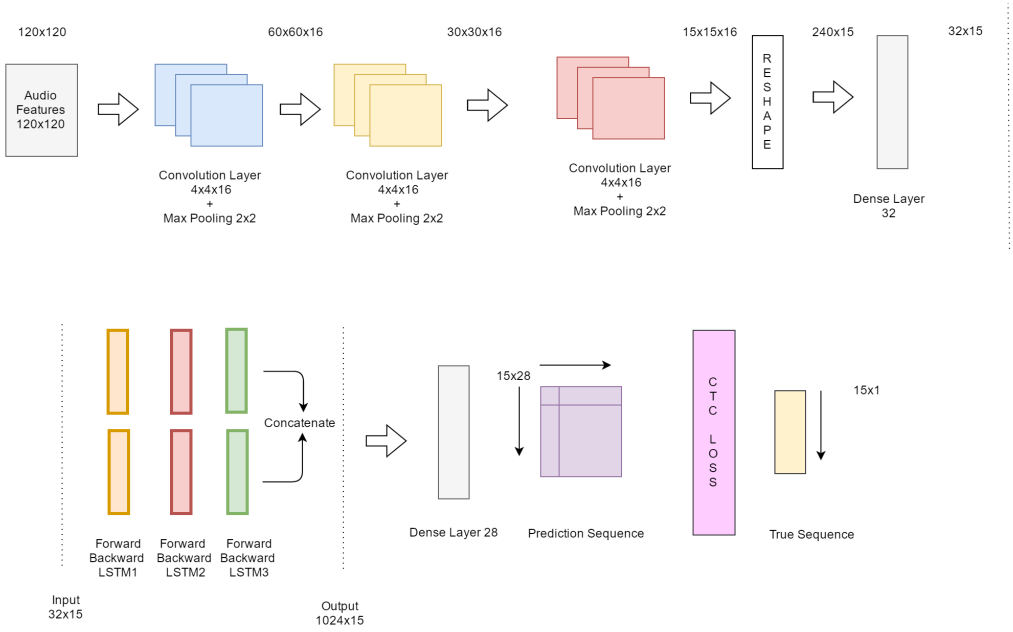


Figure 7: Audio Neural Network

Audio Network

Above,we explained how the relevant audio MFCC features are extracted from the speech signal. We have a time sequence of MFCC vectors with a rate of 200 MFCC feature vectors per second for 5ms hop size.Each of the features is of length 120x1.Now each spoken word is of different time length and would contain different number of MFCC feature vectors.Example,a word lasting for 300ms will contain 60 (300 divided by 5 (hop size)) features.Hence for each word in the data set, we have a speech features of 120xk where k can

vary. To bring uniformity to feature size, which is required before next step, we set size of audio features for each word to 120x120 by zero padding when k is less than 120.

This time sequence of features contains temporal speech information which needs to be modelled properly. Here,the RNN/LSTM networks come to our rescue. As we have seen earlier, RNN's are apt models for learning from temporal information. Instead of directly giving the 120x120 features to RNN, we reduce the feature size to 32x15. This is done by a 2-D convolution layer followed by max-pooling layer which down samples its input by 2.Another two sets of convolution+max pool reduces original size by 8 times. This is followed by a dense layer and the output of the layer is of size 32x15. Here the dimension 15 contains temporal information and 96 is feature size at a particular instant.

We give this 30x30 feature vector sequence as input to the bidirectional RNN's. We have 3 layers of B-RNN. This is followed by a fully connected layer which gives output of size 30x28 where 30 represents the time steps and 28 represents the classes (26 letters + 1 blank + 1 space). This is followed by CTC layer which minimizes the loss between the prediction sequence and the actual sequence to get the correct word outputs. The network architecture is shown below.

Note that the label sequence for each training signal is the word itself.This means the output of our network will be word predictions directly.

Visual Network

The visual neural network uses only the visual features described above for prediction.The network structure is very similar to the audio. In audio network we had audio MFCC features as input which were transformed via first few CNN layers in the network and then given to the LSTM. The visual network after the LSTM layer is exactly same as the audio network. In the beginning, a different set of CNN filters are applied to the video sequence. For each frame, the video feature is of size 64x64. For representing the visual features for a word, we would concatenate the 64x64 features for each frame in the duration of that word. Same as in audio case, as each word has different duration,the feature for each word is normalized to length 64x64x15.

Since we have 3-D input features, so we apply a 3-D convolution layer followed by a 3-D max pooling layer.This changes the dimension from 64x64x15 to 32x32x15x16 (due to 16 filters).Another set of same filters reduces the size

to 16x16x16x15. These neurons are reshaped to 2-D as 16^3x15.This is followed by a dense layer and output of shape 64x15 is obtained.Now these features are passed through the LSTM and the rest of network is same as before.

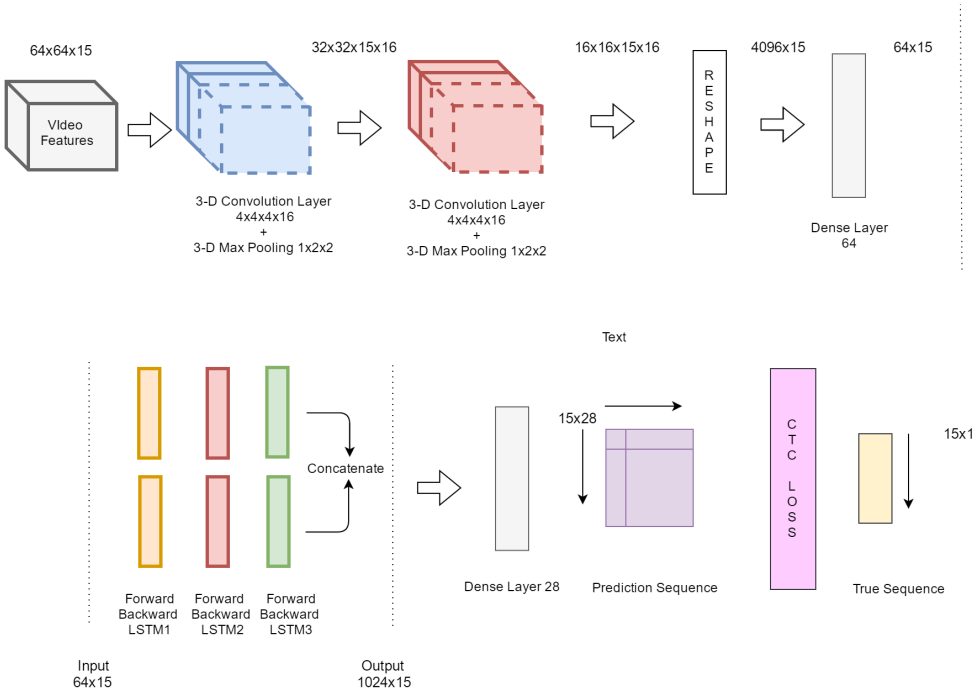


Figure 8: Video Neural Network

Audio-Visual Neural Network

Finally, we build a joint network which takes input features from the audio stream as well as the video stream and predicts the words in the speech. The idea here is that,by using a joint model, we would be able to improve upon the speech recognition results because of information from 2 modalities.

Previously, we saw separate audio and visual networks. To build a joint network, we concatenate the audio and video features before the LSTM layer and then give this combined feature vector as an input to the RNN/LSTM.Thus we combine 32x15 audio feature with 64x15 video feature to form a 96x64

feature vector.

Hence we have built here an 'end-to-end' network which means the training is done end to end without the requirement for concatenating outputs of smaller systems. Such 'end-to-end' networks make prediction of outputs using trained models simpler.

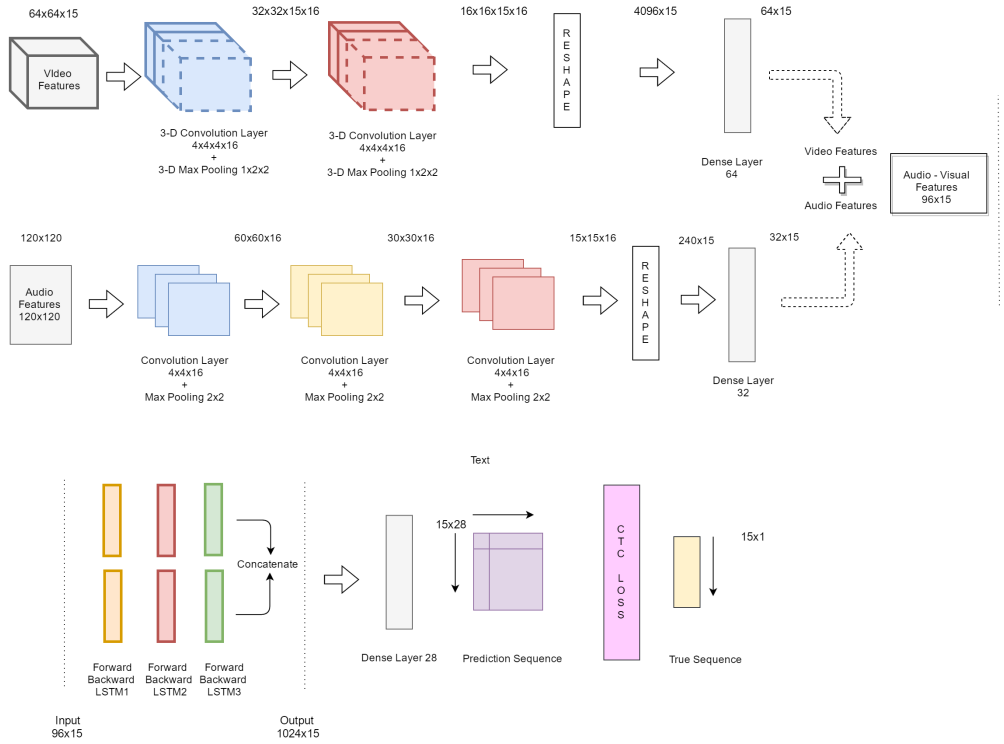


Figure 9: Audio Visual Neural Network

4 Results

We implement these models using Keras and Tensorflow deep learning libraries in python.

Dataset

In our implementation,we are using the grid corpus dataset [6] which consists

of various speakers with each person recording 1000 sentences.The length of each sentence is 6 words. The words are taken from a vocabulary of 52 words. The following table describes the set of words spoken by each person.

command	color	preposition	letter	digit	adverb
bin	blue	at	A-Z	1-9,zero	again
lay	green	by			now
place	red	in			please
set	white	with			soon

Table 1: Vocabulary in Grid Corpus Dataset [6]

Training
We train and test our network on a subset of the grid corpus dataset.We select two speakers from the dataset for training. The dataset consists of a total of **2000 sentences** and hence **12000 words** spoken by the two speakers. All the three networks are trained separately for a total of 30 iterations over the entire dataset. We use stochastic gradient descent (SGD) optimizer with the CTC loss objective.

Testing
We test our learned models on four other speakers from the dataset that are not part of training and look at predictions for the **1000 sentences** spoken by the each person. For calculating the test accuracy,we use two metrics. The first is the exact words prediction rate, where the strings are exactly matched, the other is based on nearest neighbour calculation based on *Edit Distance.*
Edit Distance is a method to check the dissimilarity between two strings by calculating the minimum number of operations required to convert one string into the other.For exactly same strings this cost in zero. There are different varients for edit distance but here we will be using what is called as the 'Levenshtein Distance' where insertion, deletion and substitution in one string is allowed for the transformation to other string.

The results for the prediction accuracy on 6000 words spoken by each speaker for all the three networks is given below:

The accuracy for the audio network serves as a baseline for our results. We can see from the prediction results that the accuracy for the network with

	Word Recognition Accuracy (Exact / Edit Distance)		
Speaker	**Audio Network**	**Visual Network**	**Audio+Visual Network**
1	84.0 / 89.3	19.0 / 24.0	**93.8 / 94.9**
2	84.0 / 90.0	30.6 / 38.5	60.4 / 67.1
3	70.7 / 79.5	37.9 / 45.8	70.2 / 75.7
4	55.2 / 63.0	25.4 / 31.8	**63.5 / 72.2**

Figure 10: Calculation of percentage accuracy for word recognition using three networks. The recognition is for 6000 words per speaker.The accuracy is calculated using two ways,(matching the exact string/ based on Edit Distance calculation)

only the visual information is very small. This result is along the expectation because prediction of speech by just looking at the speaker by judging from the lip movement is quite difficult even for a human observer. There is a many to one mapping between the spoken phone and the lip movement. So a low accuracy rate is expected here.
The result for the joint audio-visual network is better than the audio network for two speakers. This is along the lines of what we expected. By addition of visual information to the audio features, we get improved results for recognition rates. Though the results for the other two speakers do not indicate this. So, there is scope for better generalization of the model.

The following are some of the words along with corresponding predictions using the three networks.

Audio	bin'	blue'	at'	e'	nne'	bin'	bliue'	at'	e'	six'	soon'	bin'	biue'	at'	e'	seven'	please'
Video	blan'	blue'	in'	t'	now	blae'	tour'	aitn'	t'	six'	two'	blace'	twh'	at'	d'	tw'	plese'
Audio+Video	pin'	blue'	at'	e'	nyw	bin'	blue'	at'	e'	six'	soon'	bin'	blue'	at'	e'	seven'	please'
Actual	bin'	blue'	at'	e'	now	bin'	blue'	at'	e'	six'	soon'	bin'	blue'	at'	e'	seven'	please'

Figure 11: Examples of word predictions by the three networks.

5 Key Issues

Synchronization of Video and Audio streams
The Audio and the Video rates do not match which is quite expected because audio rate in samples per second is larger than video frames per second.So synchronization of the audio and video frames is an important step to make the audio and video features complement each other instead of being out of synchronization.If they are not in synch, the neural network would not train properly causing higher error rates.In our case we had audio stream at 50K samples per second and video at 25 frames per second. From audio, we extracted the MFCC vectors which reduced the rate to 200 MFCC feature vectors/sec.Therefore, the video frame rate is 1/8th of the audio feature rate.So we designed our video feature extraction neural network layers to reduce dimension size in time to match the rate for audio stream.

Video Features The performance of the AVSR system depends a lot on the quality of video features extracted and how these are combined with the audio features for training.Initially, we worked with the DCT features for video frames but the results were not very good. We realized that a better way is to use a CNN to extract the video features automatically and then give these features to the RNN. By extracting features in this way, we were able to improve the results significantly.

Tuning Neural Network Architecture
The tuning of Neural Network parameters is very important for achieving good results. This includes selecting the filter sizes in different layers, the size of features at each stage, number of neural network layers etc. These can be improved by trail and test but a good initial guess is important. Initially, we had only two layers in the Bi-RNN but later we changed it to three which improved the results.

6 Conclusion

In this work, we presented an Audio-Visual speech recognition system to analyse the performance when along with the audio information, the visual information is also taken into account.We extract relevant features from the

audio and video streams and compare performance of three networks consisting of audio, visual and audio-visual information respectively. We produce our results on small parts of a larger dataset as a proof of concept. We observe that the performance of just visual network is not upto mark. But the audio-visual network created by concatenating the audio and visual features can improve the performance over just audio based networks which is observed for two speakers.

7 Future Work

- The accuracy we get for the AVSR system is still less as compared to the best results. The result for combined audio and visual streams depend on the quality of features representing the data.Though the MFCC features that we obtain for the audio part are quite standard, the results for AVSR depend a lot the quality of the video features that we obtain. Though we experimented with Discrete Cosine Transform (DCT) features, the results were not very good. We have used a CNN to extract out the features from the video stream. To improve on the existing results, the need is to explore and identify the different ways in which most relevant video features can be extracted and utilized.

- Once we have useful audio and video features, a lot depends on how these are combined together to train a model to get desired results. In this work, we have looked at a system where a joint network is learned on audio and visual features. For future work, we can explore different ways of multimodal training in which at different times, more focus is given to one of the two feature sets. For example, in the beginning we can train model only on video features for some time and switch off the audio channel.This helps since the audio features are more relevant as compared to video features,the network won't get biased to audio features only. After few epochs we can switch on the audio stream and train on both. This kind of discriminative training has been explored previously to prevent learning models which are biased to one particular kind of data and have been quite successful.

- Here we have looked at a pure neural network based approach for this problem. We can explore other HMM based techniques and combine them with this neural network model to improve upon existing results.

Though the neural network does not take any task specific knowledge which is one of its strengths, in certain situations we can take domain knowledge into account to build HMM models, which can help in improving accuracies.

- We looked at AVSR problem in which we have a dataset consisting of single person in a constant background environment. We performed training only on small subset of this database. An extension of this work is to train and test models on the entire dataset. A challenging problem is to extend this to the problem of 'lip reading in the wild' where we can possibly have people speaking in different background environments with possibly multiple people. The state of the art system build by Google called 'Watch, Listen ,Attend and Spell' achieves some very good results [7]. This pure deep learning based architecture can be studied and taken as a starting point for building more complex systems.

References

[1] Abhinav Thanda, Shankar M Venkatesan. *Audio Visual Speech Recognition using Deep Recurrent Neural Networks,* arXiv:1611.02879v1, 9 Nov 2016

[2] Alex Graves, Abdel-rahman Mohamed and Geoffrey Hinton. *Speech Recognition with Deep Recurrent Neural Network* IEEE, International Conference on Acoustics, Speech, and Signal Processing, 2013

[3] Alex Graves, Santiago Fernanadez, Faustino Gomezs, Jurgen Schmidhuber *Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks*

[4] Gerasimos Potamianos, Chalapathy Neti, Guillaume Gravier, Ashutosh Garg, Student and Andrew W. Senior. *Recent Advances in the Automatic Recognition of Audio-Visual Speech.* IEEE, Vol. 91, No. 9, September 2003.

[5] Ramon Sanabria, Florian Metze, Fernando De La Torre. *Robust End-to-End Deep Audiovisual Speech Recognition* arXiv:1611.06986v1 [cs.CL] 21 Nov 2016

[6] Martin Cookea, Jon Barker, Stuart Cunningham, Xu Shao *An audio-visual corpus for speech perception and automatic speech recognition*

[7] Joon Son Chung, Andrew Senior, Oriol Vinyals, Andrew Zisserman. *Lip Reading Sentences in the Wild*. arxiv:1611.05358v1, 16 Nov 2016

[8] CS231: Convolutional Neural Networks for Visual Recognition, Stanford University.

[9] http://deeplearning.net/tutorial/lenet.html

[10] http://colah.github.io/posts/2015-08-Understanding-LSTMs/

[11] http://dlib.net/face.landmark detection.py.html