LNet - Lipreading Neural Net

G001 (s2238419, s2250760)

Abstract

This study aims to classify words from the lip movement of a speaker present in the image sequence of a video without any audio signal by constructing a pipeline for Automatic Lip Reading(ALR) using MIRACL-VC1 dataset. First, we designed a pipeline that preprocesses images from the MIRACL-VC1 dataset having low-resolution image sequences to extract high-quality upscaled lip area from each image. Second, we applied a novel approach for the ALR technique by designing and implementing a CNN-LSTM + Graph Neural Network(GNN) model that learns from temporal variable in image sequence as well as captures video semantics, resulting in better learning. Our experiments done using only 2D color image sequences of the dataset achieved a classification accuracy of 64% on unseen speaker data instances, surpassing the current best model with 56% accuracy on this dataset that used only 2D color image sequences. Our accuracy is also comparable to the current highest accuracy of 63% achieved by a model trained on a combination of both 2D color image sequences and depth image sequences of this dataset to the best of our knowledge.

1. Introduction

Automatic Lip Reading(ALR) is a way of inferring the speech content based on just the speaker's lip movement, that is, without using the audio signal. This is based on real-world techniques where people with hearing impairment communicate using sign language. ALR can help tackle many challenges like overcoming the difficulty of learning gesture language or getting speech content from videos that either do not have an audio signal or have a very noisy audio signal(Hao et al., 2020). Nowadays, video traffic is increasing at a very high rate due to the internet and a techniques like ALR can help extract data and gain insights from such videos like profanity detection from sporting events. But, implementing an ALR system has its challenges. The fundamental limitation is one caused by homophones, that is, sets of words that sound different like park, mark, and bark but have identical lip movement and cannot be distinguished by lip-reading without context(Chung & Zisserman, 2016). Other difficulties are caused due to variations like different accents, speed of speaking, and mumbling as well as effects that deteriorate image quality like bad lighting, resolution, and shadows.

Various approaches have been taken to solve this problem. In the study Lip Reading Word Classification (Abiel Gutierrez), different deep learning models including CNNs, RNNs, and existing publicly available pre-trained networks are used for the task of word classification using the 2D color image sequences from the MIRACL-VC1 dataset(Rekik et al., 2014). The highest accuracy achieved on the seen(speaker dependent) test dataset is 59% whereas, on the unseen(speaker independent) test dataset, the average accuracy achieved is 10%. In the study (Rekik et al., 2014), the use of Support Vector Machines(SVMs) is investigated for the task of word as well as phrase classification on the MIRACL-VC1 dataset as well. They have used both 2D color image sequence and depth image sequence for training and prediction based on different types of features extracted from images like Histogram of Oriented Gradients(HOG) and Motion Boundary Histograms(MBH). The best accuracy achieved on unseen test data without using depth image sequences is 48.1% and for seen data is 92.8%. The same with a combination of both depth images and 2D color images is 63.1% and 95.3% respectively. In the study, Lip reading using CNN and LSTM, various deep learning methods like CNN model from scratch, a CNN-LSTM model, and pre-trained VGGNet model are investigated (Amit Garg). The best result achieved in this study is by concatenating the sequence of images into a single image and using these images with pretrained the VGGNet model. The accuracy achieved using this approach on unseen data is 56%.

The task of this study is to create an end-to-end pipeline for word classification from a sequence of video frames. This study aims to create an ALR system that can provide inference from data that is similar to that of real-world scenarios. Specifically, this study uses low-resolution images that require multiple preprocessing steps for enhancement of the face and lip area. As opposed to (Chung & Zisserman, 2016) where they created a highly preprocessed dataset, we aim to work with a dataset that is crude so that the inference can be done from similar crude videos already available like traffic camera footage

Our work differs from the existing work mentioned in this study in two ways. First, our pipeline preprocesses the input image sequence which consists of the whole person and not just the face to extract high-quality upscaled lip area from a low-resolution image sequence, thereby removing the dependency of having a highly preprocessed dataset and still achieving better results. Second, we have

trained a CNN-LSTM model on the preprocessed data. After this model is trained, instead of using it for inference, the second last layer of this trained network, representing the hidden latent representation of each image sequence, is extracted and used as node embeddings to train a Graph Neural Network(GNN). We used GNN to capture the semantic relationships between independent image sequences which other studies have not considered for this application.

2. Data set and task

For this study, we have used the MIRACL-VC1 dataset for the task of words classification(Rekik et al., 2014). This dataset contains 1500 labeled video instances for words as well as 1500 labeled video instances for phrases. For our task, we only consider the video instances of words. There are a total of 10 unique words, listed below in table 1, uttered in the dataset by a total of 15 speakers, each speaking each word 10 times, giving a total of 1500 data instances. Each video instance has 2 types of frame sequences, one is of 2D color images and another is of depth images acquired by a Kinect Sensor with a resolution of 640 x 480 and frame rate of 15 fps. For this study, only the sequence of 2D color images are considered as collecting depth images is not feasible for every application. For example, using this system to infer words from a silent movie clip.

Class	Word
01	Begin
02	Choose
03	Connection
04	Navigation
05	Next
06	Previous
07	Start
08	Stop
09	Hello
10	Web

Table 1. Word List of MIRACL-VC1 Dataset (Rekik et al., 2014)

The reason for choosing this dataset is that the images in this dataset are not heavily preprocessed beforehand. For example, the Oxford-BBC Lip Reading in the wild(LRW) dataset consists of all videos that are 29 frames long, each image is focused on the face of the speaker and all the images are of high quality (Chung & Zisserman, 2016). As this study focuses on building an ASR system that aims to work on real-world inputs, it is not always possible to get a high-quality input and hence, in this sense MIRACL-VC1 dataset is a better representation of real-world scenarios. A comparison between MIRACL-VC1 and LRW dataset is shown in Figure 1.

As mentioned above, the main task of this study is classification of words from a sequence of frames of a video. We consider the concept of speaker-dependent and speaker-independent test sets. A speaker-dependent test set is constructed from speakers which have been featured in the train

Figure 1. MIRACL-VC1 vs LRW (Rekik et al., 2014) (Chung & Zisserman, 2016)

set as well. A speaker-independent test set is constructed from a speaker which has not been featured in the training set. For this study, we divide the 1500 data instances into three sets: a speaker-independent test set with 100 instances from a single speaker, a train set with 1300 instances from speakers excluding the one featured in the test set, and a speaker-dependent validation set with 100 instances, featuring the same speakers as that in the training set. This study uses a speaker-independent test set because it shows how the model performs on unseen data, again, giving a better representation of real-world scenarios.

Certain preprocessing steps are required on the given dataset before making changes to each 2D color image. First, as only words are used for this study, video instances related to phrases are deleted. Then, depth image sequences from each word are removed. Then, A new class dataset is created which contains 10 classes, each representing a unique word and containing 150 instances(10 utterances of that particular word from 15 speakers). In the end, this class dataset is split into a speaker-independent test set, a train set, and a speaker-dependent validation set as explained above. Test and validation set each will have 10 classes and each of these classes will have 10 video instances. The train set will also have 10 classes, where each class will have 130 instances.

3. Methodology

In this section, we describe the details of our proposed solution LNet: Lipreading Neural Net Figure 2 to detect the words spoken based on silent video as an input. Entire proposed architecture can be divided into two main subsections (i) Video Pre-processing (ii) Video Classification.

3.1. Video Pre-Processing

Before the image sequences are fed to the model, multiple image processing steps are applied which enhances the quality of the image, thereby generating better data to learn from. The first step in image preparation is using the Dlib library to extract faces from the original image as only lips contribute to the learning for this study(dli, a). DLIB is an

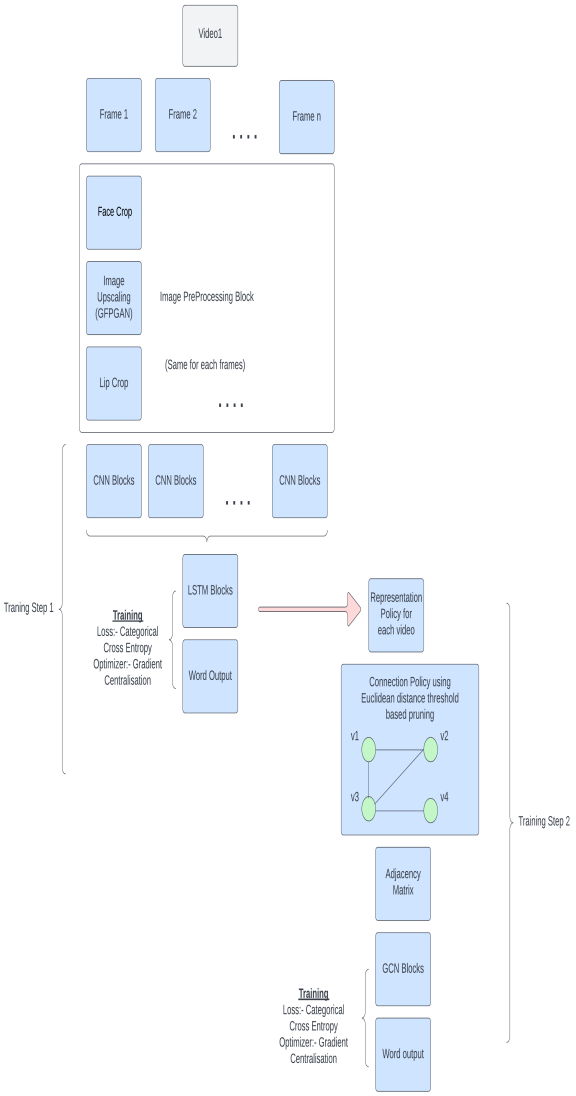

Figure 2. LNet Architecture.

open-source library that has software components to aid in solving many tasks including Machine Learning(dli, b). Specifically, in this study, DLIB's facial landmark detection feature is used to extract the speaker's face from the original image. This feature allows a computer to extract landmarks of different regions of the face like eyes, mouth, nose, and eyebrows as well as the whole face. The extracted and cropped face is fixed to a size of 90x90 because increasing its resolution further leads to distortion of the image.

The end images to be fed to the model are of lips because ALR is based on learning from lip movement to infer the speech content. Extracting lips from a 90x90 face image results in a very small sized image. Zooming into this image(which is expected by a deep neural network for better performance) degrades the quality of the image, as shown in Figure 4 which deteriorates the training of the model. To tackle this problem, in this study, a face restoration framework called GFPGAN is used(gfp). The GFPGAN model is trained on the FFHQ dataset which contains 70,000 face images at 1024×1024 resolution and contains considerable variation in terms of age, ethnicity, and image background(ffh). For this study, each cropped face image is enhanced using GFPGAN's pretrained v1.3 model(gfp). The results improved the quality of each cropped face image considerably and in turn, made it possible to create high-quality lip images. The enhanced cropped face image is of 148x150 resolution.

Then, the DLIB library is again used to extract lip area by generating mouth landmarks. These images are fixed to 60x40(median cropped lip image size in the dataset) size because a Convolutional Neural Network model needs images of fixed size. An example of such a resulting image as well as a complete sequence of frames for one data instance can be seen Figure 3 and Figure 5 respectively.

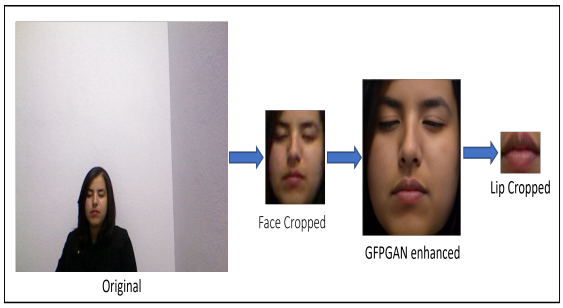

Figure 3. Image during different stages of transformation

Figure 4. Frame sequence without GFPGAN enhancement

Figure 5. Frame sequence with GFPGAN enhancement

After this step, there are 1500 data instances, i.e, a sequence of frames of the lip area, consisting of 1300 training instances, 100 test instances, and 100 validation instances. Having only 1300 training instances, i.e, only 130 instances per word are less for training. To increase the training data, video augmentation is done in this study using the Vidaug python library (Vid). This library provides effective video augmentation techniques for training a Convolutional Neural Network. It provides a total of 30 types of possible augmentation methods. For this study, 4 of these are used, namely, add, horizontal rotate, random rotate, and random translate. The augmentation is done only on the training data. After applying these 4 augmentations to each training instance and adding the resulting instances to the original

training set, a training set of total of 6500 instances is constructed. An example of various augmentations is shown in Figure 6.

Figure 6. Different Augmentations

Another factor to consider is the varying number of frames for each video instance. This number of frames should be fixed before it can be used for either training or inference. In this study, a VariableLengthGenerator class is created that generates a batch of data given a batch size, input, and output. Fixing of frame size is done using padding with zero matrices of size 60x40. For example, one video instance has 7 frames and the maximum sequence length from the available video instances is 25, 18 zero matrices are added before the 7 frames of the first video and the result is both videos having a fixed frame size of 25. But, it is possible that for this study such long sequences are rare, and padding all video instances using it leads to higher memory consumption and more redundant data. So, the VariableLengthGenerator samples data from supplied input and output based on batch size and pads each video sequence based on the maximum sequence length for that batch and not the whole training set, thereby reducing both redundant data and memory consumption. Test and validation sets are also processed using this VariableLengthGenerator to convert it to the same data format as that of the training set. But, the batch size is passed as one, so no padding is applied and individual test instances are received from the generator.

3.2. Video Classification

Several Video classification methods, as in the case of images, have been proposed with Convolution Neural Networks are successful as well. But one of the major issue with such architectures is that they skip the learning from important temporal variable that exists in videos as sequence of frames. Several architectures like CNN-LSTM (Donahue et al., 2014) or ConvLSTM (Shi et al., 2015) or optical flow (Simonyan & Zisserman, 2014) have been proposed to utilize the existing temporality in videos. However all of these methods do not consider the video semantics that represents the contextual relationships between different video instances (or data points itself), which can be used in assisting the training process. In order to capture this gap, our proposed classifier is using Graphical Neural Networks (GNN) in addition to a CNN-LSTM base architecture(Costa et al., 2021).

Video Classifier takes the pre-processed video input of cropped lips and then gives out the prediction probabilities of each possible word. The pipeline of the proposed GNN classifier can be majorly subdivided into three blocks (i) Representation Policy (ii) Connection Policy (iii) Graphical Convolution Layers (GCN). Considering the graph structure in our case each video represents a node for which node embeddings are created using representation policy and its connection to other nodes are decided using connection policy. Also because of the lesser data instances, which is also one of the challenges in current setting as well as an ideal expectation in real life, model usually gets over-fitted easily. We will also discuss the specific training settings (optimizers, regularizers and dropouts) used so as to make training stable and model generalization better under given data constraint.

3.2.1. Representation Policy

This section of our classifier is responsible for generating a single vector representation for each video. To generate such embeddings, we trained a CNN-LSTM model Figure 7 in Training step 1 Figure 2. The CNN-LSTM model uses a series of CNN and pooling blocks to extract the spatial features from each frames which is then passed through the series of LSTM blocks to learn from temporal variable that exists in video (sequence of frames). Once we have trained this model against correct target word, we used second last layer as our final embeddings for each node Figure 7.

Figure 7. Representation Policy

3.2.2. Connection Policy

After generating the node embeddings using representation policy, we used our connection policy to decide which nodes are connected through an edge and also edge weights as another continuous confounding edge variable. There are several possible connection strategies that can be used. Two extreme strategies could be self connection loop i.e, connecting each nodes just to itself but it would not be effective in learning contextual relations or connecting all nodes which would require high memory usage. We have used a better trade off connection policy of threshold based edge pruning (Costa et al., 2021). We calculated the euclidean distance between each pair of nodes (1) and then connected the pair of nodes by an undirected edge if the distance between them is less than average euclidean distance between all possible pairs of nodes(2). Once the nodes are connected based on this policy, each of the formed edges are given weights as inverse of the distance between that particular pair of nodes (3) and then finally they are normalized as well. (4)

$$d_{i,j} = \sqrt{(\mathbf{v_i} - \mathbf{v_j})^2} \tag{1}$$

$$e_{i,j} = \begin{cases} 1 & d_{i,j} <= d_{avg}, \\ 0 & d_{i,j} > d_{avg} \end{cases} \tag{2}$$

$$w_{i,j} = \frac{1}{d_{i,j}} \tag{3}$$

$$w^N_{i,j} = \frac{w_{i,j}}{\Sigma_j w_{i,j}} \tag{4}$$

3.2.3. Graphical Convolution Network

Once we have generated the node embeddings, determined the edges and edge weights, we used 2 layers of graphical convolution neural networks to further update the node embeddings based on neighbouring nodes so as to understand the video semantics (contextual relations between different videos). We have implemented our Graph Convolution Layer as a custom keras layer following Design Space for graph neural networks (You et al., 2020). So our graph convolution layer performs the following 3 steps.

Prepare: A pre-processing layer applying a Feed Forward Neural Network (FFN) to the initial node embeddings to generate a message.

Aggregate: The message thus generated for each nodes are then combined with the neighbouring nodes based on weights assigned to each of the edges using some permutation invariant pooling operations (max, sum or mean) to generate a final aggregated node embeddings. This aggregated representation contains contextual information of each video.

Update: The initial node embeddings and the aggregated node embeddings both with the shape num nodes X embedding dim are then finally combined together and passed through an another FFN to generated the final node embeddings for each node. We have implemented and tried out three combination rule starting with the two primitive strategies of concatenation and adding. We then also implemented a GRU based combination strategy where both initial and aggregated node embeddings are stacked horizontally to form a single sequence which is then passed through a GRU layer.

Then this final node embeddings generated is passed through a softmax layer to generate each word prediction probabilities.

3.2.4. Training settings

Given the very less amount of data points for training compared to other datasets like LRW (Chung & Zisserman, 2016), training was very unstable and model overfitted very easily. Thus in order to improve the generalisation performance we used the l1 l2 regularizers, dropout layers and learning rate schedulers where we start with a very high learning rate of 0.1 and if validation performance doesn't change for a certain patience value (epochs) we decrease the learning rate by a factor of 10. But apart from all these conventional approaches, we have added a specific functionality to our existing optimizer called Gradient Centralisation (GC) as proposed in (Yong et al., 2020) to further improve our generalisation performance. Inspired by the effect of batch normalisation on activations and weight regularizer on weights, GC tries to normalise the gradients directly.

Gradient Centralisation: It is a simple yet effective optimisation techniques for all Deep Learning based models. It can help in obtaining better generalization performances of Deep Learning based models specially when there are lesser number of training instances with the help of weight space and output feature space regularisation (Yong et al., 2020). Results that have been mentioned in the literature shows the improvements on using this optimizer with different types of models Resnet, Xception or Densenets.

This normalisation also makes the training process more stable by improving the loss function and its gradient Lipschitzness. It even makes sure to avoid gradient explosion (similar characteristics as that of gradient clipping).

Mathematically, It just normalises the gradient vector to have zero mean. For a multidimensional layer, mean of the gradient is calculated across each dimension and is then subtracted from the gradient across the same dimension i.e, GC operator is defined as

$$\phi_{GC}(grad(\mathbf{w_i})) = grad(\mathbf{w_i}) - \mu_{grad(\mathbf{w_i})} \tag{5}$$

$$\mu_{grad(\mathbf{w_i})} = \frac{1}{M}\Sigma_{j=1}^{M} grad(\mathbf{w_{i,j}}) \tag{6}$$

where $grad(\mathbf{w_i})$ = refers to gradient of loss function with respect to i^{th} column vector of weight matrix,

4. Experiments

In this section, we will describe the details of experiments that we have carried out. The best performance that we have achieved is after applying GNN as a classifier using latent representation learned by CNN+LSTM blocks (representation policy) on top of video just using cropped lip images. These videos were also upscaled by GFPGAN before lips cropping. We have used classification accuracy as the performance metrics for all different methods we tried out. These accuracies for both speaker independent and dependent settings are mentioned in Table 2. Apart from classification accuracies, we have mentioned detailed confusion matrix and learning curve for our best performing model using GNN in Figure 8, Figure 9 and Figure 10.

Methods	**SD**	**SI**
Without GFPGAN	67	23.99
With GFPGAN	81.63	33.33
With GFPGAN + Video Augmentation	77.55	57
With GFPGAN + Video Augmentation + Graph Neural Network *	85.71	64

Table 2. Accuracy in % for difference model variants tried out.

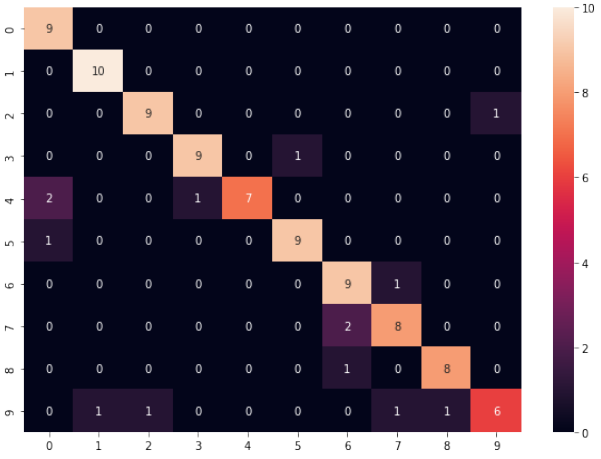

Figure 8. Confusion matrix for speaker dependent

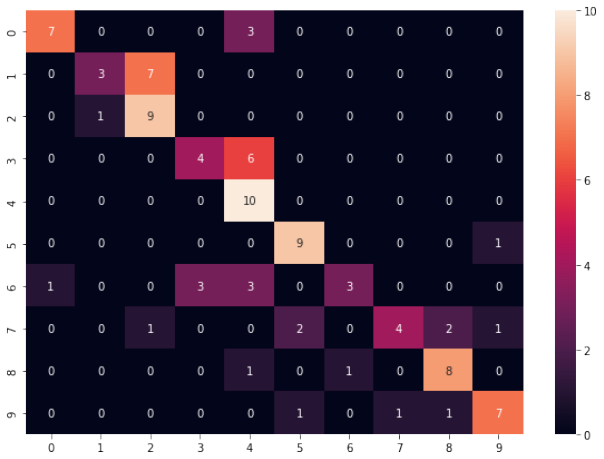

Figure 9. Confusion matrix for speaker independent

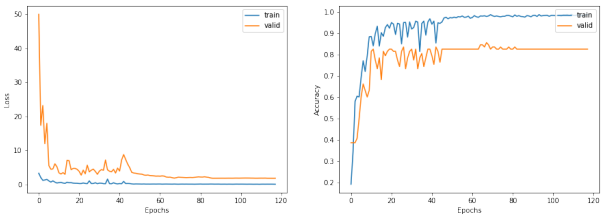

Figure 10. Learning curve for GNN final model

5. Related work

We compare our proposed architecture LNet with the existing state of art methods and few other attempts to enhance it in Table3.We have used the classification accuracy as the performance metrics for comparison

Methods	**SD**	**SI**	**Average**
(Abiel Gutierrez)	59	10	34.5
(Amit Garg)	73	56	64.5
(Rekik et al., 2014)	**92***	48.1	70.05
Ours	85.71	**64***	**74.855***

Table 3. Accuracy comparison for different models in literature

As observed from Table3, we have clearly surpassed the performance metrics of previous methods in literature with a good margin. All papers we have surveyed are using only color images for this dataset. (Rekik et al., 2014) is the only paper which has used depth images as well and they got around 4% higher average accuracy than us because of their better performance in speaker dependent scenario. But even after using depth images, they achieved a speaker independent accuracy of 63% which is close to what we have achieved 64% from our model. By only using color images we have achieved 5% higher average accuracy and in the case of speaker independent we have achieved 16% higher accuracy than them. Furthermore, capturing depth images in real life is not feasible as it requires a dedicated device to capture such images.

6. Conclusions and Future Work

From the experiments that have been carried out in this study, we can conclude that our novel approach achieved the current state-of-the-art speaker independent accuracy of 64% using just 2D color images. It has also been observed that using GNN resulted in better learning for this application and for our few data instances setting, it increased the classification accuracy by 7%. Also, using GFPGAN face restoration framework allowed us to create upscaled high quality lip images resulting in better learning and increased accuracy. It also allowed us to perform video augmentation which was not effective earlier because of low quality images. Thus, our study shows that deep learning models can also be used to train low quality low sized datasets which

are a better representation of real-world scenarios.

Further attempts can be made to make the architecture even more robust to make it work in real-life scenario by adding a low-light image correction module in the image pre-processing block. ZeroDCE (Guo et al., 2020) can be used as a low light image enhancement technique, it works by predicting the high order tonal curves as an output which can be later multiplied across channels to enhance the image quality. Our preliminary results seem promising in Figure11 and could be pursued further in the development of Robust Lip reading system. Currently, we are working on a classification based task, giving out probabilities of words seen at the time of training data. The proposed architecture can be tried to train a character based model to predict out of vocabulary words as well. Instead of using a LSTM in representation policy to understand the temporal variable of a video, Attention based model can also be tried out (Chung et al., 2017). Several other Graph based architecture like Graph Attention Networks or Graph Recurrent Neural Networks can also be explored (Ward et al., 2020). We have used Gradient Centralisation as an extension to conventional optimizers, but still we have observed from the learning curve there still exists the scope of further improvement in generalisation performance and stabilisation of training process.

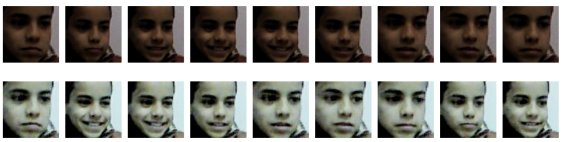

Figure 11. (i)Top images are low light images, low-light settings are simulated by decreasing the brightness in a single video.(ii) Bottom images are the output of ZERODCE we trained over our training dataset consisting of low light setting simulated images.

References

Vidaug video augmentation library. URL https://github.com/okankop/vidaug.

Dlib python library. a. URL https://pypi.org/project/dlib/.

Dlib python library. b. URL https://hectorenevarez.github.io/AIClubWorkshopsSpring21/Workshop9/DLIB.html.

Ffhq dataset. URL https://github.com/NVlabs/ffhq-dataset.

Gfpgan face restoration library. URL https://github.com/TencentARC/GFPGAN.

Abiel Gutierrez, Zoe-Alanah Robert. Lip reading word classification. URL http://cs231n.stanford.edu/reports/2017/pdfs/227.pdf.

Amit Garg, Jonathan Noyola, Sameep Bagadia. Lip reading using cnn and lstm. URL http://cs231n.stanford.edu/reports/2016/pdfs/217_Report.pdf.

Chung, Joon Son and Zisserman, Andrew. Lip reading in the wild. In *Asian conference on computer vision*, pp. 87–103. Springer, 2016.

Chung, Joon Son, Senior, Andrew, Vinyals, Oriol, and Zisserman, Andrew. Lip reading sentences in the wild. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3444–3453, 2017. doi: 10.1109/CVPR.2017.367.

Costa, Felipe, Saito, Priscila, and Bugatti, Pedro. Video action classification through graph convolutional networks. pp. 490–497, 01 2021. doi: 10.5220/0010321304900497.

Donahue, Jeff, Hendricks, Lisa Anne, Guadarrama, Sergio, Rohrbach, Marcus, Venugopalan, Subhashini, Saenko, Kate, and Darrell, Trevor. Long-term recurrent convolutional networks for visual recognition and description. *CoRR*, abs/1411.4389, 2014. URL http://arxiv.org/abs/1411.4389.

Guo, Chunle, Li, Chongyi, Guo, Jichang, Loy, Chen Change, Hou, Junhui, Kwong, Sam, and Cong, Runmin. Zero-reference deep curve estimation for low-light image enhancement. *CoRR*, abs/2001.06826, 2020. URL https://arxiv.org/abs/2001.06826.

Hao, Mingfeng, Mamut, Mutallip, Yadikar, Nurbiya, Aysa, Alimjan, and Ubul, Kurban. A survey of research on lipreading technology. *IEEE Access*, 8:204518–204544, 2020.

Rekik, Ahmed, Ben-Hamadou, Achraf, and Mahdi, Walid. A new visual speech recognition approach for rgb-d cameras. In *International conference image analysis and recognition*, pp. 21–28. Springer, 2014.

Shi, Xingjian, Chen, Zhourong, Wang, Hao, Yeung, Dit-Yan, Wong, Wai-Kin, and Woo, Wang-chun. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. *CoRR*, abs/1506.04214, 2015. URL http://arxiv.org/abs/1506.04214.

Simonyan, Karen and Zisserman, Andrew. Two-stream convolutional networks for action recognition in videos. *CoRR*, abs/1406.2199, 2014. URL http://arxiv.org/abs/1406.2199.

Ward, Isaac Ronald, Joyner, Jack, Lickfold, Casey, Rowe, Stash, Guo, Yulan, and Bennamoun, Mohammed. A practical guide to graph neural networks. *CoRR*, abs/2010.05234, 2020. URL https://arxiv.org/abs/2010.05234.

Yong, Hongwei, Huang, Jianqiang, Hua, Xiansheng, and Zhang, Lei. Gradient centralization: A new optimization technique for deep neural networks. *CoRR*, abs/2004.01461, 2020. URL https://arxiv.org/abs/2004.01461.

You, Jiaxuan, Ying, Rex, and Leskovec, Jure. Design space for graph neural networks. *CoRR*, abs/2011.08843, 2020. URL https://arxiv.org/abs/2011.08843.