Multi-View Automatic Lip-Reading using Neural Network

Daehyun Lee[1,2], Jongmin Lee[1], and Kee-Eung Kim[1]

[1] School of Computing, KAIST, Daejeon, Korea
[2] Visual Display Division, Samsung Electronics, Suwon, Korea

Abstract. It is well known that automatic lip-reading (ALR), also known as visual speech recognition (VSR), enhances the performance of speech recognition in a noisy environment and also has applications itself. However, ALR is a challenging task due to various lip shapes and ambiguity of visemes (the basic unit of visual speech information). In this paper, we tackle ALR as a classification task using end-to-end neural network based on convolutional neural network and long short-term memory architecture. We conduct single, cross, and multi-view experiments in speaker independent setting with various network configuration to integrate the multi-view data. We achieve 77.9%, 83.8%, and 78.6% classification accuracies in average on single, cross, and multi-view respectively. This result is better than the best score (76%) of preliminary single-view results given by ACCV 2016 workshop on multi-view lip-reading/audio-visual challenges. It also shows that additional view information helps to improve the performance of ALR with neural network architecture.

1 Introduction

Human understands speech from not only acoustic information but also visual clues. An extreme case is shown in the McGurk effect [1], a visual information of /ga/ with an acoustic of /ba/ is perceived as /da/. Similar phenomenon also appears in machines using deep learning approach [2]. These observations present that visual information gives significant clues to the speech recognition in both human and machine.

Automatic lip-reading (ALR), also known as visual speech recognition (VSR) or speech reading, is understanding speech using only visual information (movements of the lips, face, tongue and so on) without acoustic information. In audio-visual speech recognition (AVSR), ALR has an important role in enhancing the performance of speech recognition through audio-visual data fusion. Furthermore, stand-alone ALR is a useful component of various applications such as a visual password, silent speech interface, and forensic video analysis.

ALR performance is measured by word or phrase classification error rate of utterance in the form of lip image sequence. ALR is a challenging task because of many visual factors such as various mouth shapes, changes of illumination, and head poses variation [3].

There are several types of experimental setups in ALR task; speaker dependent (SD) or speaker independent (SI), single-view or multi-view. In SD setting, one speaker data are used for both training and evaluation. On the other hand, evaluation is performed to the unseen speaker(s) in SI setting. In the multi-view setting, we use the data recorded from the multiple cameras with various angles simultaneously [4] while the data in a single-view setting is recorded from one camera angle.

Classical approaches are based on visual feature extraction methods, such as principal component analysis, discrete wavelet transform, discrete cosine transform, active appearance model [5], local binary pattern [6], optical flow [7], Eigenlips [8], histograms of oriented gradients [9], internal motion histograms, motion boundary histograms, and their mixed models [8, 10]. For multi-viewpoint lipreading, [11] adopt a minimum cross-pose variance analysis technique.

Thanks to recent successful achievements in deep neural network approach in machine learning community, people started to apply neural network to ALR and AVSR. For example, feature extraction has been done by deep autoencoder [2] without massive hand-crafted engineering work. Recently, [12] show that with end-to-end neural network architecture, they can achieve a state-of-the-art performance in speaker dependent setting, compared to classical feature-based approaches.

In this paper, we tackle the ALR task using an end-to-end neural network approach without hand-crafted feature extraction in multi-viewpoint SI setting experiment. Our work motivated by recent works of video and image recognition researchers that use convolutional neural network (CNN) with long short-term memory (LSTM) for understanding video and image data [13–15].

2 Background

2.1 Problem Specification

We deal with three ALR tasks given by ACCV 2016 workshop on multi-view lip-reading/audio-visual challenges[1] (MLAC 2016) as follows:

- Single-view ALR: Train and test on data recorded from a single camera view.
- Cross-view ALR: Learn and transfer knowledge from a source view (e.g., the frontal view) to enhance learning for a target view (e.g., the profile view).
- Multiple-view ALR: Train and test on synchronized data recorded from multiple camera views.

2.2 Convolutional Neural Network

CNN is a biologically inspired variant of multi-layer perceptron containing small sub-regions of a visual field called receptive field [16]. Unlike fully connected layered network, CNN has sparse connectivity and shared weights for the purpose

[1] http://ouluvs2.cse.oulu.fi/ACCVW.html

of increasing computational efficiency and global representation power. CNN is now the most popular and effective selection for learning visual features in computer vision and machine learning fields.

We obtain a feature map at layer h with input x pixel at coordinates (i, j) as the following equation:

$$h_{ij} = a((W * x)_{ij} + b) \tag{1}$$

, where weight matrix W and bias vector b is the filter of this feature map, a is activation function for non-linearities.

2.3 Long Short-Term Memory

Recurrent neural network (RNN) is designed for processing sequential data by sharing weights across several time steps. Due to its vanishing gradient problem that appears to long-sequence training data, its variations including LSTM [17] become popularized in practical applications. LSTM consists of memory cells connected recurrently to each other, which is replacing hidden units of standard RNN. End-to-end learning architecture with LSTM is the typical model when dealing with a sequence dataset.

We update LSTM hidden state h_t at every timestep t as follows:

$$\begin{aligned} i_t &= \sigma(W_i x_t + U_i h_{t-1} + b_i) \\ f_t &= \sigma(W_f x_t + U_f h_{t-1} + b_f) \\ c_t &= i_t * tanh(W_c x_t + U_c h_{t-1} + b_c) + f_t * c_{t-1} \\ o_t &= \sigma(W_o x_t + U_o h_{t-1} + V_o c_t + b_o) \\ h_t &= o_t * tanh(c_t) \end{aligned} \tag{2}$$

, where x_t is an input at time t and W_i, W_f, W_c, W_o, U_i, U_f, U_c, U_o, V_o are weight matrices, b_i, b_f, b_c, b_o are bias vectors, subscripts represent input(i), forget(f), cell(c) and output(o) variables [18, 19].

3 Dataset

3.1 OuluVS2 Database

OuluVS2 database, publicly provided by CMVS[2], contains video recordings from 52 speakers with five different camera views at the same time. It has three collections of data - ten continuous digit strings, ten daily-use short English phrases, and five randomly selected TIMIT sentences.

OuluVS2 provides region of interest (ROI) videos, which were preprocessed by segmenting individual utterances and cropping off ROIs, for digit strings and phrases collection. We use visual part of phrases collection, which are same as

[2] The Center of Machine Vision Research, Department of Computer Science and Engineering, University of Oulu, Finland

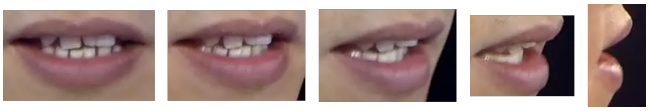
Table 1. Data separation according to speakers. The test set is given by MLAC 2016, validation set is randomly chosen as the same size of test samples. (Total 7,800 samples with 52 speakers, ID 1 to 53 except 52.)

Data Set	Speaker IDs	#Samples
Training	1,2,3,10,11,12,13,18,19,20,21,22,23,24,25, 27,33,35,36,37,38,39,45,46,47,48,50,53	4,200
Validation	4,5,7,14,16,17,28,31,32,40,41,42	1,800
Test	6,8,9,15,26,30,34,43,44,49,51,52	1,800

the preliminary experiments conducted by MLAC 2016[3]. Every phrase was uttered three times in this collection, the total number of samples is $52(speakers) \cdot 5(views) \cdot 3(utterances) \cdot 10(phrases) = 7,800$, where maximum length of utterance is 36.

The preliminary experiments use three different methods including HiLDA [20], RAW+PLVM (raw pixel values classified by latent variable models [21]). All the experiments are single-view ALR tasks, and the best result is about 76% accuracy with 45° view data and RAW+PLVM.

OuluVS2 database is more suitable for SI experiment than SD setting because of relatively large number of speakers and a small number of the utterance of each speaker. It is considerable, though out of the scope of this paper, to evaluate with a semi-SD setting that trains with multiple speakers and test unseen utterance among them.



(a) Original images with various sizes.

(b) After pre-processing with fixed size and maximized contrast.

Fig. 1. Examples of original and pre-processed lip images for a frontal, 30°, 45°, 60°, profile view image from the left to the right.

[3] http://ouluvs2.cse.oulu.fi/preliminary.html

3.2 Data Pre-Processing

First of all, we extract image frames from each ROI video by FFmpeg[4] command line tool with option *qscale:v=1*. After that, we resize all images of each view into the same size since the size of each image varies even in the same viewpoint, which enables us to conduct multi-viewpoint experiment easily. Square shape is a reasonable choice because profile view ROI image has longer height than width, unlike the others.

We have three options for pre-processing on the image itself; (1) Color: original RGB color image (2) Gray†: convert into grayscale with maximized contrast. (3) Color†: RGB color with maximized contrast.

We conduct experiments for selecting both size and image option, and the result is shown in Table 2. Experiments performed by 2D-CNN + LSTM architecture details in section 4. As a result, we use 20 by 20 pixel color image with maximized the contrast, i.e. all pixel values in each channel are normalized as mapped into [0,1] interval.

Table 2. Comiparison among various pre-processing options. All the results are obtained from validation set. Resizing into 20 by 20 pixels with contrast-maximized color image is the best option. All the later experiments are performed with this pre-processed data. †: Contrast is maximized.

Image Size	Color	Gray†	Color†	Average
25x25	75.5 %	77.1 %	76.5 %	76.4 %
20x20	77.6 %	78.6 %	**80.8** %	**79.0** %
15x15	76.8 %	74.9 %	78.2 %	76.6 %
10x10	71.8 %	75.9 %	75.1 %	74.3 %
Average	75.4 %	76.6 %	**77.6** %	

4 Proposed Method

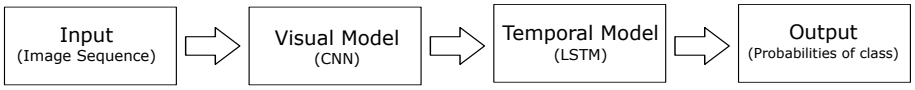


Fig. 2. Overall architecture which returns probabilities of each class from an input of image sequence of a single utterance.

We combine visual model and temporal model based on CNN, LSTM for the basic neural network architecture in Fig.2. We define visual models with

[4] https://ffmpeg.org/about.html

various settings as 2D-CNN and 3D-CNN for automatic feature extraction. We also define temporal model as LSTM that learns the temporal features of the image sequence and classify the utterance into a phrase.

4.1 Visual Model

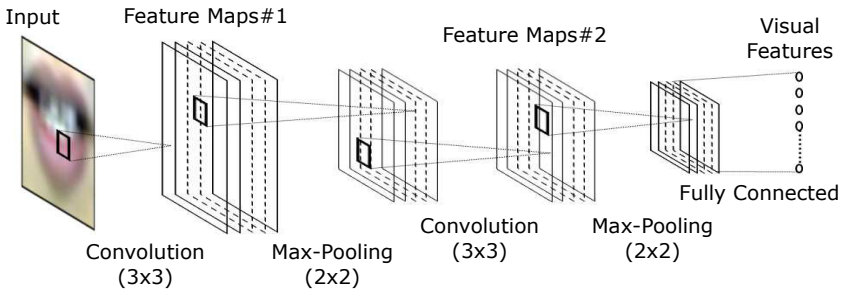


Fig. 3. 2D-CNN Architecture

2D-CNN Two convolutional layers with 16 to 256 filters in the shape of (3,3) used for feature extraction of lip region images. Each convolutional layer has a successive max-pooling layer for downscaling by the shape of (2,2). Last one fully connected layer with 8 to 64 dimension outputs used to the final output of image feature. We adopt dropout [22] technique for improving generalization power of the network. We use hyperopt [23] to find the optimal hyper-parameters (number of filters, output dimension of the fully connected layer) for our network.

3D-CNN In the same way of equation (1), we generate the feature map from a 3D input where the x pixel at 3-dimensional coordinates (i, j, k) as following equation:

$$h_{ijk} = a((W * x)_{ijk} + b) \tag{3}$$

We construct 3D-CNN as almost same as 2D-CNN, the filter shapes are (1,3,3) and (2,3,3) for the first and the second convolutional layer respectively as shown in Fig.4. (a). The shape of pooling size is (1,2,2) for every max-pooling layer, and the remaining parameters are same as 2D-CNN. Strictly speaking, the five view data is not a 3-dimensional data. However, we conduct an experiment with 3D-CNN to find out the possibilities of learning some features or not.

Merge Channels We make an input image having 15 channels from the five view data with three (RGB) channels as shown in Fig.4. (b). The places in the same pixel position of each view data are the different locations in actual. Although it looks somewhat weird, we conduct this experiment as the same reason as 3D-CNN.

Merge Images As shown in Fig.4. (c), we append five images from the different view at the same time into a single image as an input of the visual model. In this architecture, we expect to learn all the five view feature by 2D-CNN. While out of our experiment, a more elaborate configuration is that all five images avoid convolving each other along the edges.

Merge Features The last variation is shown in Fig.4. (d). We merge the features that are generated by 2D-CNN from the images in the different view at the same time. We design this architecture to learn each view image separately and to combine them into a single feature for an input of the temporal model.

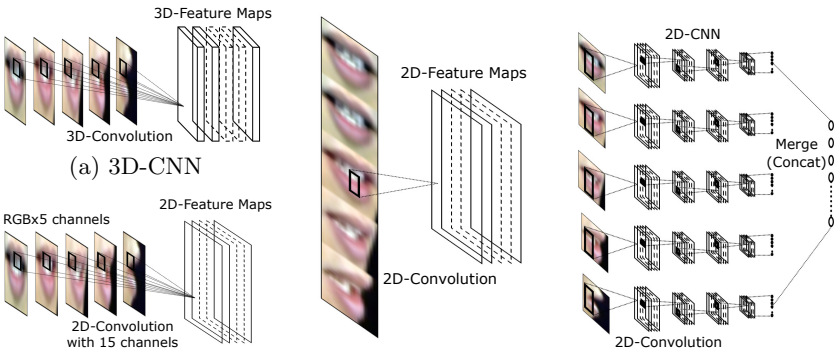


Fig. 4. Various visual model suggestions for the multi-view setting.

4.2 Temporal Model

We design the temporal model as a classifier that has inputs generated by the visual model and an output as probabilities of each class. It consists of two successive LSTM layer with 128 memory cells that lead to one FC layer having the same number of class, 10 output dimension. We select 128 by experiments using hyperopt which yield lowest validation loss between 8 to 256.

5 Experiments and Results

Basic protocol of experiments is SI setting, and the data is divided into the train, validation, and test sets as shown in Table 1. In order to improve generalization performance, we add dropout [22] layers between the layers. Learning is performed by Adam [24] optimizer with default learning rate 0.001, and categorical cross entropy objective loss function used. We evaluate the result on test data with the weights having minimum validation loss during the training until 200

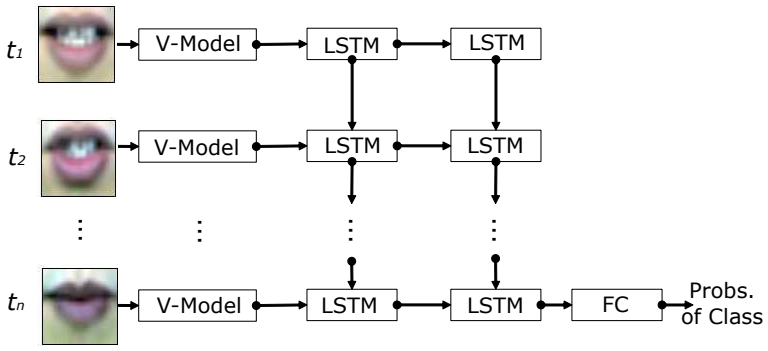


Fig. 5. Unfolded view of temporal model architecture, It returns probabilities of each class with inputs as $t_1, t_2, ..., t_n$ images in a single utterance. V-Model: Visual Model.

epoch. All experiments are performed by deep learning framework Keras [25] with Theano backend [19].

5.1 Single-View ALR

Table 3. Single-view word test accuracy results for each camera view. The best performance is obtained by profile view. †: Approximated accuracies from the graph image of preliminary single-view experiment given by MLAC 2016

	Our Results (Accuracy of Test Data)					
Training Data	(1)Frontal	(2)30°	(3)45°	(4)60°	(5)Profile	Average
(1)Frontal	81.1 %					
(2)30°		80 %				
(3)45°			76.9 %			77.9 %
(4)60°				69.2 %		
(5)Profile					82.2 %	

	Single-View Baseline Results† (Accuracy of Test Data)					
Method	(1)Frontal	(2)30°	(3)45°	(4)60°	(5)Profile	Average
DCT-PCA-HMM†	63%	62%	62%	63%	57%	61%
DCT-HiLDA-HMM†	74%	72%	73%	73%	68%	72%
RAW-PLVM†	73%	75%	76%	75%	70%	74%

Table 3. represents the single-view experiment result on test data. The baseline accuracies are given by MLAC 2016 as a preliminary experiments[5] in the form of a graph. The accuracies are approximated from the chart.

[5] http://ouluvs2.cse.oulu.fi/preliminary.html

The average accuracy is 5% higher than the average accuracy using RAW-PLVM method which is the best in the baseline. Moreover, all accuracies except 60° view are higher than the best score (76%) of the single-view baseline.

Our neural network produces the reasonable result of the prediction. The confusing phrases are similar to those of human. "Thank you" and "See you" pair is the most confusing as presented by confusion matrix in Fig.6.

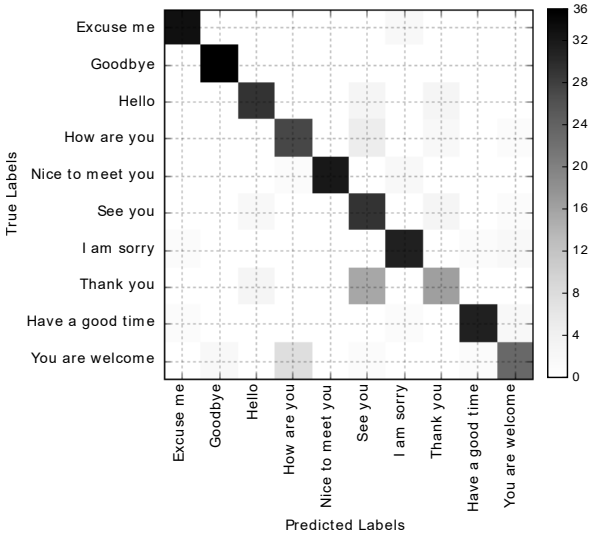


Fig. 6. An example of the confusion matrix. (profile view result with test accuracy: 82.2%)

5.2 Cross-view ALR

As the first stage (cross-view), we conduct a cross-view experiment by training all the data in train set. In other words, we train frontal, 30°, 45°, 60°, profile view data all together, and test each view separately. In this approach, we get the average accuracy 82.6%, and all of the results are better than preliminary results. We infer from this result that neural networks learn and transfer knowledge from a view to the other view by simply learning with the data altogether without further work.

In the second stage (cross-view2), we perform ALR tasks with each view data using the neural network initialized by the weights from the first stage result, as similar as fine-tuning. We choose the weights for testing with min-validation-loss, note that there is no improvement with 30°, 45° view data. As a final result, we get 83.8% in average and 86.4% in the best (profile view), and this is outperformed the preliminary results. The confusion matrices of profile view show that the progress of increasing performance as shown in Fig.7.

Table 4. Cross-view ALR test accuracy and validation-loss results for each camera view. CV (cross-view): Train all the training data mixed, CV2 (cross-view2): Train each view data individually after train all the training data mixed.

		Accuracy of Test Data					
	Training Data	(1)Frontal	(2)30°	(3)45°	(4)60°	(5)Profile	Average
CV	All	80.6 %	81.1 %	85 %	82.5 %	83.6 %	82.6 %
CV2	All+(1)Frontal	82.8 %					83.8 %
	All+(2)30°		81.1 %				
	All+(3)45°			85 %			
	All+(4)60°				83.6 %		
	All+(5)Profile					86.4 %	

		Loss of Validation Data					
	Training Data	(1)Frontal	(2)30°	(3)45°	(4)60°	(5)Profile	Average
CV	All	0.4372	0.4957	0.4109	0.3391	0.5606	0.4487
CV2	All+(1)Frontal	0.3216					0.424
	All+(2)30°		0.4957				
	All+(3)45°			0.4109			
	All+(4)60°				0.3371		
	All+(5)Profile					0.5546	

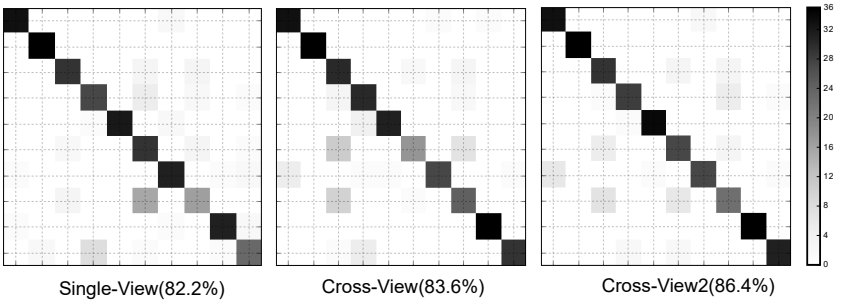


Fig. 7. The confusion matrices of test accuracies in profile view setting. It shows that a gradual improvement from single-view to cross-view. The x-axis is prediction classes; the y-axis is true classes with the same label in Fig.6.

This result indicates that the test performance of target view has been improved by transferring knowledge of other source view data. For instance, in the profile view, the accuracy is 82.2% in single-view (train profile view data only) then 1.4% improved in cross-view (train five view data altogether). Finally, we get 86.4% accuracy in cross-view2 (initialize weights by the cross-view result and train profile view data once more) which is 4.2% higher than single-view.

5.3 Multiple-view ALR

Table 5. Multi-view results of prediction accuracy using various architectures.

Method	Test Acc.
3D-CNN (MV-3D)	76.1 %
Merge Channels (MV-MC)	78.9 %
Merge Images (MV-MI)	80.0 %
Merge Features (MV-MF)	79.4 %

We conduct four types of architecture explained in section.4. In this experiments, we use hyper-parameters already tuned in the single-view experiment because we have not enough time to optimize each network one by one. Despite we adopt the hyper-parameters from single-view experiments, the multi-view result is better than the single-view result in average.

It infers that the more features (five times more than the single view) in a single frame help the network in the average performance.

On the other hand, the result is worse than cross-view in spite of using same data. The difference between them is the number of samples, the numbers of training sample are 4,200 (cross-view) vs. 840 (multi-view). We know that the number of samples impacts the neural network performance, more data is always better.

Table 6. The average computational times (minutes) of training data for 200 epochs. We use 1 GPU (Geforce GTX 1080) for each experiment.

Method	Time	#Samples	Input Dimension
Single-view	6	840	$20 \cdot 20 \cdot 3 = 1,200$
Cross-view	30	4,200	$20 \cdot 20 \cdot 3 = 1,200$
3D-CNN (MV-3D)	38	840	$5 \cdot 20 \cdot 20 \cdot 3 = 6,000$
Merge Channels (MV-MC)	7	840	$20 \cdot 20 \cdot 15 = 6,000$
Merge Images (MV-MI)	20	840	$100 \cdot 20 \cdot 3 = 6,000$
Merge Features (MV-MF)	23	840	$(20 \cdot 20 \cdot 3) \cdot 5 = 6,000$

Table 6. shows the elapsed time in the average of each experiment. We use 1 GPU, and measure the time for 200 epoch. Cross-view experiment takes exactly 5 times more time than single-view because of the same architecture and the training data size. Note that the elapsed times in the multi-view are various in the range between 7 and 38 depending on the architecture. 3D-CNN takes 5.4 times more time than Merge Channels.

All of the experiment results are summarized in Fig.8.

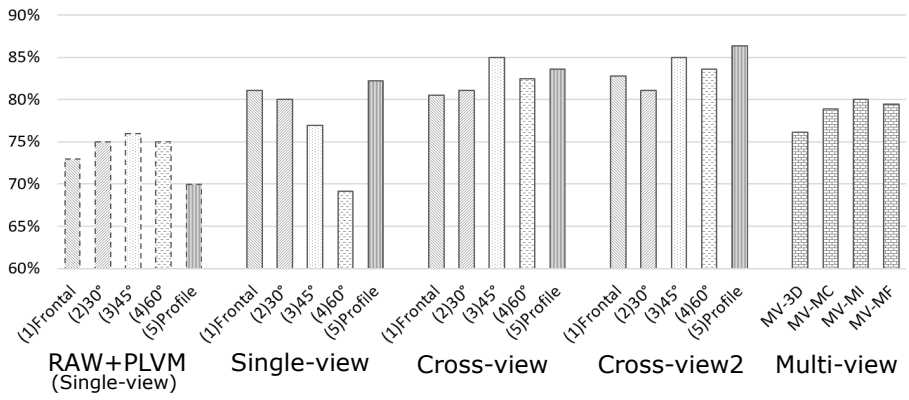


Fig. 8. The summary graph of the prediction accuracies. RAW+PLVM is the best result of the preliminary single-view experiment given by MLAC 2016. MV-3D: 3D-CNN, MV-MC: Merge Channel, MV-MI: Merge Images, MV-MF: Merge Features.

6 Conclusions

In this paper, we propose an end-to-end neural network architecture for speaker independent ALR task with multi-view data (OuluVS2 database). The evaluation is performed by experiments in single-view, cross-view, and multi-view setting. We report that the accuracies are 77.9%, 83.8%, and 78.6% in average; 82.2%, 86.4%, and 80.0% in the best respectively. All of the results are better than the best single-view result (76%) of the baseline. These results show that ALR performance is improved by multi-view data using neural network architectures.

We expect that the best score would come out in the multi-view result before experiments are performed, but the results are different as we reported. One of the reason is the lack of samples in the multi-view experiment in spite of more complex feature than a single view. Experiments with various data augmentation from multi-view data are future work.

OuluVS2 database provides the original high-resolution videos, experiment with this large size of the input video is another considerable future work.

Acknowledgement. This work was supported by Institute for Information & communications Technology Promotion(IITP) grant funded by the Korea government(MSIP) (No.B0101-16-0307, Basic Software Research in Human-level Lifelong Machine Learning (Machine Learning Center)).

References

1. McGurk, H., MacDonald, J.: Hearing lips and seeing voices. Nature **264** (1976) 746–748
2. Ngiam, J., Khosla, A., Kim, M., Nam, J., Lee, H., Ng, A.Y.: Multimodal deep learning. In: Proceedings of the 28th international conference on machine learning (ICML-11). (2011) 689–696
3. Potamianos, G., Neti, C.: Audio-visual speech recognition in challenging environments. In: INTERSPEECH. (2003)
4. Anina, I., Zhou, Z., Zhao, G., Pietikäinen, M.: Ouluvs2: a multi-view audiovisual database for non-rigid mouth motion analysis. In: Automatic Face and Gesture Recognition (FG), 2015 11th IEEE International Conference and Workshops on. Volume 1., IEEE (2015) 1–5
5. Cootes, T.F., Edwards, G.J., Taylor, C.J., et al.: Active appearance models. IEEE Transactions on pattern analysis and machine intelligence **23** (2001) 681–685
6. Zhao, G., Barnard, M., Pietikainen, M.: Lipreading with local spatiotemporal descriptors. IEEE Transactions on Multimedia **11** (2009) 1254–1265
7. Shaikh, A.A., Kumar, D.K., Yau, W.C., Azemin, M.C., Gubbi, J.: Lip reading using optical flow and support vector machines. In: Image and Signal Processing (CISP), 2010 3rd International Congress on. Volume 1., IEEE (2010) 327–330
8. Bregler, C., Konig, Y.: eigenlips for robust speech recognition. In: Acoustics, Speech, and Signal Processing, 1994. ICASSP-94., 1994 IEEE International Conference on. Volume 2., IEEE (1994) II–669
9. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). Volume 1., IEEE (2005) 886–893
10. Rekik, A., Ben-Hamadou, A., Mahdi, W.: A new visual speech recognition approach for rgb-d cameras. In: International Conference Image Analysis and Recognition, Springer (2014) 21–28
11. Pass, A., Zhang, J., Stewart, D.: An investigation into features for multi-view lipreading. In: 2010 IEEE International Conference on Image Processing, IEEE (2010) 2417–2420
12. Wand, M., Koutn, J., et al.: Lipreading with long short-term memory. In: 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE (2016) 6115–6119
13. Venugopalan, S., Rohrbach, M., Donahue, J., Mooney, R., Darrell, T., Saenko, K.: Sequence to sequence-video to text. In: Proceedings of the IEEE International Conference on Computer Vision. (2015) 4534–4542
14. Venugopalan, S., Xu, H., Donahue, J., Rohrbach, M., Mooney, R., Saenko, K.: Translating videos to natural language using deep recurrent neural networks. arXiv preprint arXiv:1412.4729 (2014)
15. Yao, L., Torabi, A., Cho, K., Ballas, N., Pal, C., Larochelle, H., Courville, A.: Describing videos by exploiting temporal structure. In: Proceedings of the IEEE International Conference on Computer Vision. (2015) 4507–4515

16. Hubel, D.H., Wiesel, T.N.: Receptive fields and functional architecture of monkey striate cortex. The Journal of physiology **195** (1968) 215–243
17. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9** (1997) 1735–1780
18. Gers, F.A., Schmidhuber, J., Cummins, F.: Learning to forget: Continual prediction with lstm. Neural computation **12** (2000) 2451–2471
19. Bastien, F., Lamblin, P., Pascanu, R., Bergstra, J., Goodfellow, I., Bergeron, A., Warde-Farley, D., Bengio, Y.: Theano: new features and speed improvements. arXiv preprint arXiv:1211.5590 (2012)
20. Potamianos, G., Neti, C., Gravier, G., Garg, A., Senior, A.W.: Recent advances in the automatic recognition of audiovisual speech. Proceedings of the IEEE **91** (2003) 1306–1326
21. Zhou, Z., Hong, X., Zhao, G., Pietikäinen, M.: A compact representation of visual speech data using latent variables. IEEE transactions on pattern analysis and machine intelligence **36** (2014) 1–1
22. Srivastava, N., Hinton, G.E., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. Journal of Machine Learning Research **15** (2014) 1929–1958
23. Bergstra, J., Yamins, D., Cox, D.D.: Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. ICML (1) **28** (2013) 115–123
24. Kingma, D., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
25. Chollet, F.: Keras. https://github.com/fchollet/keras (2015)