The 14th International Conference on Auditory-Visual Speech Processing
25-26 August 2017, Stockholm, Sweden

Exploring ROI size in deep learning based lipreading

Alexandros Koumparoulis,[1] *Gerasimos Potamianos,*[1,2] *Youssef Mroueh,*[3] *Steven J. Rennie*[3]

[1] Electrical and Computer Eng. Dept., University of Thessaly, Volos 38221, Greece
[2] Athena Research and Innovation Center, Maroussi 15125, Athens, Greece
[3] IBM T. J. Watson Research Center, Yorktown Heights, NY 10598, U.S.A.

alkoumpa@uth.gr, gpotam@ieee.org, {mroueh,sjrennie}@us.ibm.com

Abstract

Automatic speechreading systems have increasingly exploited deep learning advances, resulting in dramatic gains over traditional methods. State-of-the-art systems typically employ convolutional neural networks (CNNs), operating on a video region-of-interest (ROI) that contains the speaker's mouth. However, little or no attention has been paid to the effects of ROI physical coverage and resolution on the resulting recognition performance within the deep learning framework. In this paper, we investigate such choices for a visual-only speech recognition system based on CNNs and long short-term memory models that we present in detail. Further, we employ a separate CNN to perform face detection and facial landmark localization, driving the ROI extraction process. We conduct experiments on a multi-speaker corpus of connected digits utterances, recorded in ideal visual conditions. Our results show that ROI design choices affect automatic speechreading performance significantly: the best visual-only word error rate (5.07%) corresponds to a ROI that contains a large part of the lower face, in addition to just the mouth, and at a relatively high resolution. Noticeably, the result represents a 27% relative error reduction compared to employing the entire lower face as the ROI.

Index Terms: lipreading, speechreading, visual speech recognition, region-of-interest, CNN, LSTM, deep learning.

1. Introduction

Lately, there has been renewed research interest in automatic speechreading (or lipreading) systems, harvesting recent advances in the computer vision and automatic speech recognition (ASR) fields, driven by rapid progress in deep learning [1]. Such systems exploit visual speech information, extracting it from video of the speaker's face, and employ it, often fusing it with the audio stream, for ASR. The technology promises to facilitate robust human-computer interface development and processing of vast video data archives, among others [2, 3].

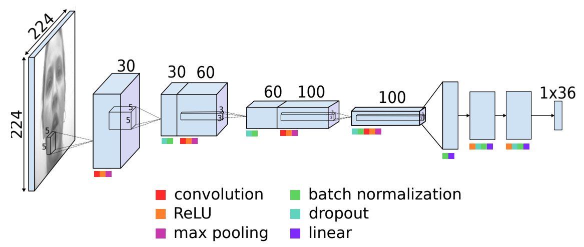

Figure 1: *A schematic of the regression CNN employed for face detection and facial landmark localization in Section 2.1 (see also Table 1).*

Visual feature extraction is a crucial component of speechreading systems, while also shared by a number of related audio-visual processing tasks [3, 4]. Appearance-based methods have dominated its design, employing appropriate representations of the video data within a region-of-interest (ROI), containing the speaker's mouth, and extracted following face detection and facial landmark localization [3, 5]. Traditionally, hand-crafted features have been used for this purpose, mostly based on PCA, image transforms such as DCT, wavelets, and scattering [6], or image descriptors like LBPs [7] and HOGs [8]. Although such features have recently been employed in conjunction with deep learning methods for visual speech modeling [9, 10, 11], it is the use of convolutional neural networks (CNNs) that has dominated over the past couple of years, yielding state-of-the-art speechreading systems [12, 13, 14, 15, 16]. In this approach, the entire ROI constitutes the CNN input layer, and a hierarchy of layers of numerous convolution filters together with other appropriate operations are learned, removing the need for hand-crafted feature design. Furthermore, motivated by recent progress in ASR [17, 18, 19], CNNs have been primarily incorporated into speechreading systems in conjunction with long short-term memory (LSTM) models [20] and their variants [21], in order to achieve temporal speech modeling [8, 15, 16, 22, 23, 24, 25]. Most such works implement so-called "end-to-end" systems, based on connectionist temporal classification (CTC) [26] or sequence-to-sequence learning [27], successfully addressing data sequence labeling.

As the aforementioned works operate on the ROI raw data, it is reasonable to expect that ROI extraction, including its physical coverage and resolution may affect system performance. This has certainly been demonstrated within the traditional speech modeling paradigm in our earlier work [28, 29]. However, little or no attention has been paid in the literature to

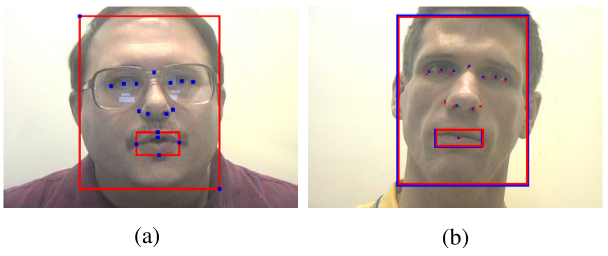

Figure 2: *(a) The 18 points predicted by the regression CNN (marked in blue). The corresponding face and mouth bounding boxes are drawn in red. (b) Face and mouth detection results on a frame of the dataset of Section 4.1 (shown in red), compared to ground truth annotations (shown in blue). Additionally, 12 localized landmarks are depicted.*

10.21437/AVSP.2017-13

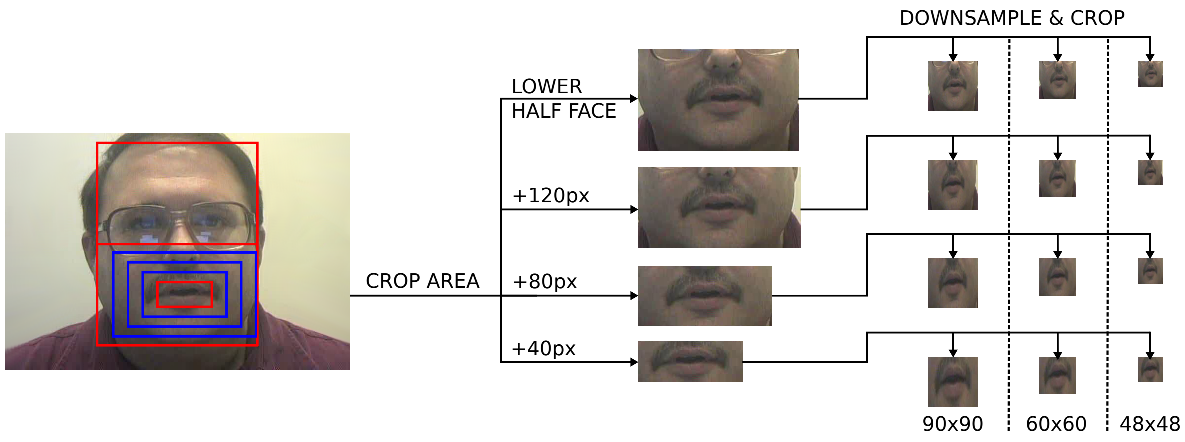

Figure 3: *The ROI extraction process following face and mouth detection, depicted on an example dataset frame. Most of the ROIs considered in our experiments are also shown, differing in resolution and physical coverage of the lower face around the mouth.*

this issue within the CNN-LSTM framework adopted by the recently proposed state-of-the-art speechreading systems. In this paper, we investigate the problem based on our CNN-LSTM system that we also present in detail. The system also employs a separate CNN to perform very accurate face detection and facial landmark localization, driving the ROI extraction process. Reported experiments are conducted on an in-house multi-speaker corpus of connected digits utterances, recorded in ideal visual conditions [30].

The remainder of the paper is structured as follows: Section 2 discusses the ROI extraction process; Section 3 the CNN-LSTM system and decoding framework; Section 4 the data and experimental results; and Section 5 our conclusions.

2. Face detection and ROI extraction

2.1. CNN based face detection and landmark localization

The first step in our visual-only ASR pipeline is, given a video frame, to localize 16 facial landmarks and another two for face bounding box detection. For this purpose, we train a regression CNN of four convolutional and three fully connected layers, using Adagrad optimization [31] and the smooth L_1 loss function [32], based on annotated images (see Section 4.1). The CNN architecture is schematically depicted in Figure 1 and further detailed in Table 1. The input to the network is a grayscale image, where we rescale each pixel intensity from [0,255] to the [0,1] range and apply zero mean and unity variance normalization [33]. Notice that the database frames have a 704×480-pixel resolution, and are converted to 224×224 pixels before fed into the CNN. Specifically, we crop the frames to 672×448 pixels (removing image boundary pixels) and then downsample them by 3×2 along the x- and y-axes, obtaining 224×224-pixel frames. Figure 2 depicts the 16 + 2 points that the CNN is trained to detect, together with a detection example.

2.2. Landmark post-processing and ROI extraction

The CNN predictions (obtained independently at each frame) are temporally smoothed using median filtering over a 15-frame window. This step rejects any outliers and results in smoother predictions over time. From the filtered facial landmarks on each frame, we calculate the average mouth bounding box and the mouth center. The mouth bounding box results from the four outer mouth landmarks, as shown in Figure 2.

We enlarge the initial mouth bounding by adding space around it at a ratio of 3:2 along the x- and y-axes, due to the initial frame size (704×480 pixels). We experiment with three physical coverage scales, adding 40, 80, and 120 pixels to the initial bounding box along the y-axis (half at each side; the corresponding increments along the x-axis are 60, 120, and 180 pixels, respectively, due to the 3:2 ratio mentioned). Further, a fourth case is considered, consisting of the lower half face, as returned from the face bounding box. The four cases will be denoted as “+40px”, “+80px”, “+120px”, and “LHF”. Corresponding ROIs are cropped from the input frame and typically resampled at three resolutions, namely 96×96, 64×64, and 48×48 pixels (additional resolutions are considered in the +80px case – see Section 4.2). In the first two cases, we select the inner 90×90 and 60×60 pixels, respectively, to allow model training with bigger batch size, thus better utilizing the available GPU resources and reducing training time. The process is illustrated in Figure 3.

The aforementioned scheme does not take camera-subject distance and face size into account. However, due to the relative small variation in the corpus (see Section 4.1), there is little variance in the resulting ROI size compared to the mouth size. In particular, the ROI physical width to mouth width ratio has mean (standard deviation) of 1.37 (0.023), 1.74 (0.046), 2.12 (0.069), and 2.18 (0.090), over the database, for the +40px, +80px, +120px, and LHF cases, respectively. Clearly, in case of more unconstrained visual data, the ROI extraction scheme can be easily modified to ensure constant such ratio in each considered case.

3. CNN-LSTM system architecture

3.1. CNN for visual feature extraction

The CNN architecture used for visual feature extraction is schematically depicted in Figure 4 and further detailed in Table 2. All convolutional layers operate on each video frame independently. The last layer (“concat”) operates on a window of 3 consecutive frames with stride one. We do not add any padding to the sequence: as a result, the LSTM input sequence is 2 time-steps shorter than the original frame sequence. While the architecture shown in Table 2 is used for all ROI sizes, we replace the two 1x3x3 max pooling layers with 1x2x2 max pool-

Layer	Output Size	Parameters
Spatial convolution	$30 \times 113 \times 113$	5x5F, 3x3S
ReLU	$30 \times 113 \times 113$	
Spatial max pooling	$30 \times 56 \times 56$	3x3F, 2x2S
Spatial dropout	$30 \times 56 \times 56$	
Spatial batch norm.	$30 \times 56 \times 56$	
Spatial convolution	$60 \times 56 \times 56$	5x5F, 2x2S
ReLU	$60 \times 56 \times 56$	
Spatial max pooling	$60 \times 27 \times 27$	3x3F, 2x2S
Spatial dropout	$60 \times 27 \times 27$	
Spatial batch norm.	$60 \times 27 \times 27$	
Spatial convolution	$100 \times 27 \times 27$	3x3F, 1x1S
ReLU	$100 \times 27 \times 27$	
Spatial max pooling	$100 \times 13 \times 13$	2x2F, 2x2S
Spatial dropout [34]	$100 \times 13 \times 13$	
Spatial batch norm.	$100 \times 13 \times 13$	
Spatial convolution	$100 \times 13 \times 13$	3x3F, 1x1S
ReLU	$100 \times 13 \times 13$	
Spatial max pooling	$100 \times 6 \times 6$	3x3F, 2x2S
Batch norm. [35]	3600	
Linear	512	
ReLU	512	
Dropout [36]	512	
Batch norm.	512	
Linear	512	
ReLU	512	
Dropout	512	
Batch norm.	512	
Linear	36	

Table 1: *Details of the regression CNN used for face detection and facial landmark localization (see also Figure 1). The following notation is used: F: convolutional filter size; P: feature map padding size; S: stride size.*

ing layers when $\min\{\text{ROI width}, \text{ROI height}\} \leq 32$ pixels. A smaller input also shrinks the output of the last max pooling layer to size $T \times 120 \times 1 \times 1$; as a result, we also change the size of the input to the linear layer, whenever necessary. No other changes are applied to the CNN. Similarly to the regression CNN of Section 2.1, we rescale the pixel intensities $[0, 255] \rightarrow [0, 1]$, and apply zero mean and unity variance normalization. Calculating the mean and variance of pixels over a video dataset can easily result in numerical overflow. To avoid this, we calculate a pixel histogram for each frame, divide it with its sum and accumulate all histograms into one. Finally, we divide the accumulation histogram with its sum, before calculating the mean and variance.

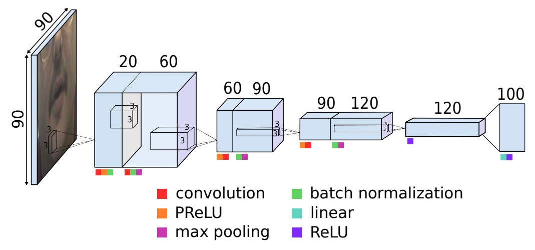

Figure 4: *A schematic of the CNN employed for visual feature extraction in Section 3.1 (see also Table 2).*

Layer	Output Size	Parameters
Vol. convolution	$T \times 20 \times 45 \times 45$	1x3x3F, 1x2x2S
PReLU [37]	$T \times 20 \times 45 \times 45$	
Vol. batch norm.	$T \times 20 \times 45 \times 45$	
Vol. convolution	$T \times 60 \times 45 \times 45$	1x3x3F, 1x1x1S
Vol. batch norm.	$T \times 60 \times 45 \times 45$	
Vol. max pooling	$T \times 60 \times 15 \times 15$	1x2x2F, 1x2x2S
PReLU	$T \times 60 \times 15 \times 15$	
Vol. convolution	$T \times 90 \times 15 \times 15$	1x3x3F, 1x1x1S
Vol. batch norm.	$T \times 90 \times 15 \times 15$	
Vol. max pooling	$T \times 90 \times 5 \times 5$	1x3x3F, 1x3x3S
PReLU	$T \times 90 \times 5 \times 5$	
Vol. convolution	$T \times 120 \times 5 \times 5$	1x3x3F, 1x1x1S
Vol. batch norm.	$T \times 120 \times 5 \times 5$	
Vol. max pooling	$T \times 120 \times 2 \times 2$	1x2x2F, 1x2x2S
ReLU	$T \times 120 \times 2 \times 2$	
Linear	$T \times 100$	
ReLU	$T \times 100$	
Concat	$T \times 3 \times 100$	$t-1, t, t+1$

Table 2: *Details of the CNN used for visual feature extraction (see also Figure 4). Output sizes correspond to a $T \times 1 \times 90 \times 90$ input. The same notation is used as in Table 1.*

3.2. LSTM for temporal modeling

The basic block of our LSTM network is the resLSTM shown in Figure 5. It consists of two LSTM networks (one forward and one backward) of 300 units each. The activations of the two networks are summed, and the sum is passed through a batch normalization layer that operates independently on each frame. Our resLSTM differs slightly from the one introduced in [38]. There, a residual connection is employed, but with a convolutional LSTM and without a batch normalization layer. A convolutional LSTM might be beneficial to lipreading (since it would preserve the spatial structure of the input signal), but it is very computationally demanding (compared to a non-convolutional LSTM); for this reason we didn't consider it.

Our LSTM network consists of four resLSTM layers, followed by a linear (projection) layer that operates independently on each frame and has dimensions 19×300 (19 being the number of target classes, as discussed in Section 3.3). The network is depicted in Figure 6.

The LSTM layers are implemented using the CUDNN library [39], which is highly optimized for GPU computation. One main difference in the CUDNN LSTM implementation

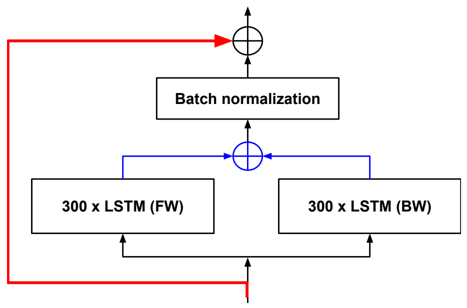

Figure 5: *The resLSTM block diagram. The residual connection is denoted by the red arrow, and the summation of activations by the blue arrows. The two operations are applied independently per time-step. The block includes a forward LSTM (left) and a backward LSTM (right), both with 300 units.*

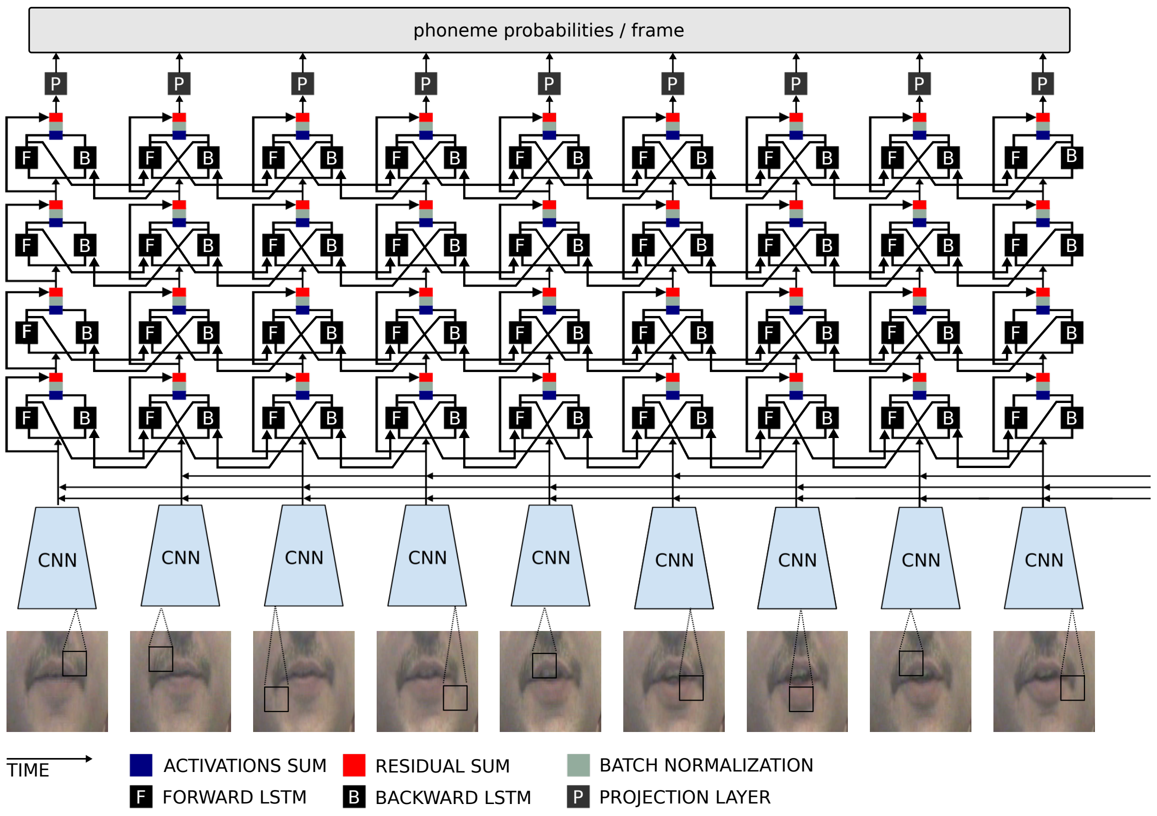

Figure 6: *The CNN-LSTM system depicted for 9 time-steps.*

from [40] is the use of two bias terms and the lack of peephole connections. We apply dropout (0.3) to the LSTM layers. The equations describing this specific implementation for a unidirectional LSTM layer are:

$$\begin{aligned} i_t &= \sigma_i(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_{W_{xi}} \\ &\quad + b_{W_{hi}}) \end{aligned} \tag{1}$$

$$\begin{aligned} f_t &= \sigma_f(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_{W_{xf}} \\ &\quad + b_{W_{hf}}) \end{aligned} \tag{2}$$

$$\begin{aligned} c_t &= f_t \odot c_{t-1} + i_t \odot tanh_c(W_{xc}x_t + W_{hc}h_{t-1} \\ &\quad + b_{W_{xc}} + b_{W_{hc}}) \end{aligned} \tag{3}$$

$$\begin{aligned} o_t &= \sigma_o(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_{W_{xo}} \\ &\quad + b_{W_{ho}}) \end{aligned} \tag{4}$$

$$h_t = o_t \odot tanh_h(c_t) \tag{5}$$

where σ is the logistic sigmoid function and i_t, f_t, c_t, o_t are respectively the input gate, forget gate, cell activations, and the output gate. We initialize both bias terms of the forget gate to unity, as suggested in [21].

3.3. CTC loss and decoding

The network is trained using the CTC loss function [26]. We use 18 phonemes plus one reserved as the blank label, required by the CTC function. First, we convert all corpus words (eleven digits – see Section 4.1) to their corresponding phoneme sequence, using the CMU pronouncing dictionary [41]. Then, we construct a tree using all eleven phoneme sequences, where each tree branch represents a phoneme, each leaf a vocabulary word, and a path from the tree root to a leaf a phoneme sequence. The tree is depicted in Figure 7.

Assuming a video with T frames, at every time-step the network outputs a probability distribution over the 19 labels. As a result, the output for the entire video will be $T \times 19$. From this output, at every frame we select the most probable phoneme, and we create a new phoneme sequence of size $T \times 1$. From this sequence we remove all repeated labels and then all blank labels, resulting in a new sequence of size $N \times 1$, where $N \ll T$. Starting from the first phonemes of the sequence, we traverse the tree until we find a leaf node. If during traversal there is no branch for the current head of the sequence, we discard the sequence head and repeat for the next phoneme. We convert all utterances to their corresponding phoneme sequence, and train the network on this sequence, thus we do not directly minimize the word error rate (WER) but the phoneme error rate [17].

This approach is by no means robust to single phoneme errors, scalable to larger vocabularies, or computationally optimal. Nonetheless, given the size of the problem considered, we obtain very good results overall, as discussed in Section 4.2. A better solution would be the well established option of a sequence-to-sequence model [27] in an encoder-decoder fashion with the attention mechanism [42]. Such architectures are able to model language and spelling within a single framework. Also it has the advantage of being able to incorporate multimodal signals [15]. Other options for the decoder include hybrid architectures, such as RNN-HMMs [43] or RNN-WFSTs [44]. For small-vocabulary problems, a simple beam search with a language model can also improve WER [26].

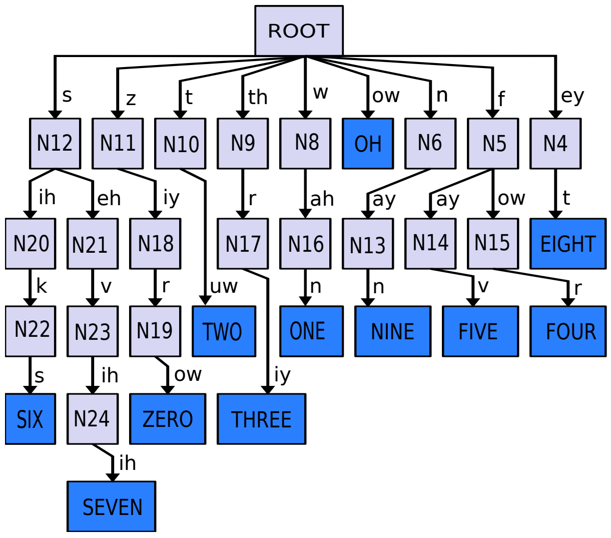

Figure 7: *The tree structure discussed in Section 3.3, representing the 11-digit dictionary of our small-vocabulary corpus.*

4. Data and experiments

4.1. Database

For our experiments, we use the IBM audio-visual database of connected digits, recorded under ideal, "studio"-like, visual conditions [30]. The corpus contains 50 subjects and approximately 6.7k utterances of 7- or 10-digit strings ($\sim$ 10 hours in duration). The data vocabulary consists of 11 words, i.e., "one"–"nine", "zero", and "oh". Video is recorded at 30 Hz and a 704×480-pixel resolution, as already mentioned.

We adopt a multi-speaker training/testing experimental paradigm, partitioning the corpus at a 77.4 / 22.6% ratio between training and test sets. Further, for regression CNN based face detection and landmark localization, we have an available pool of 5446 manually annotated frames (see also Figure 2). This set is split at a 79 / 21% ratio for regression CNN training and testing.

4.2. Results

To evaluate the performance of the regression CNN, we calculate precision and recall of both face and mouth bounding boxes, compared to their respective ground truths, according to the well-known formulas:

$$\text{Prec}(G_i, P_i) = \frac{Area(G_i) \cap Area(P_i)}{Area(P_i)}$$

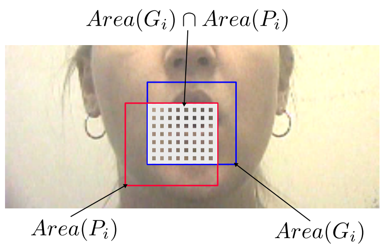

Figure 8: *Benchmarking mouth detection performance, as discussed in Section 4.2.*

frame size	Physical ROI size expansion			LHF
	+40px	+80px	+120px	
90×90	6.886	5.913	5.505	6.964
60×60	5.872	5.072	6.008	–
48×48	6.256	6.640	6.557	–
32×32	–	7.589	–	–
24×24	–	7.935	–	–
16×16	–	8.702	–	–

Table 3: *Visual-only WER (%) achieved by the presented CNN-LSTM system for various ROIs differing in physical coverage and resolution. Additional resolutions are considered for the best performing coverage setting (+80px).*

$$\text{Rec}(G_i, P_i) = \frac{Area(G_i) \cap Area(P_i)}{Area(G_i)}$$

where G_i denotes the ground truth and P_i the predicted bounded box for frame i (see also Figure 8 for the case of the mouth bounding box). Then, face and mouth detection F-scores can be calculated as:

$$\text{F-score}(G_i, P_i) = \frac{2\ \text{Prec}(G_i, P_i)\ \text{Rec}(G_i, P_i)}{\text{Prec}(G_i, P_i) + \text{Rec}(G_i, P_i)}$$

Evaluated over the test part of the manually annotated frames (1143 frames), the face detection F-score is 0.969 and the mouth detection one is 0.838 (balanced recall and precision are observed). These results are quite satisfactory, and help drive the good performance of the speechreading system, reported next.

We now proceed to visual-only recognition results, presented in WER. As evidenced by Table 3, ROI physical size and resolution affect WER significantly. The best result of 5.07% WER is obtained by the +80px ROI at a 60×60 resolution. Results degrade when ROI extraction attempts to cover less (+40px) or additional parts (+120px) of the lower face. Interestingly, in the latter case, a better result is achieved by the higher resolution ROI (90×90 pixels) vs. the 60×60 one. This is probably due to the fact that the system "focuses" on the more central parts of the ROI at a resolution similar to the best operating point of the +80px setting. Few more ROI resolution values are considered in the +80px setting, demonstrating the benefits of high-resolution video input. Interestingly, mouth ROIs down to a 16×16-pixel size still contain significant lipreading information, but of course degraded compared to the optimal 60×60-pixel setting. Notice also that compared to the LHF setting, the best result corresponds to a 27% relative WER reduction.

It should also be noted that the reported WERs achieved by our CNN-LSTM system represent dramatic improvements over traditional GMM-HMM based systems operating on handcrafted features. For example, a 29.5% WER was reported in [30] using a traditional approach. This highlights the major advances that deep learning methods have brought to the field.

5. Conclusions

We have demonstrated that selecting the right combination of visual ROI physical coverage and resolution has important implications to automatic speechreading system performance within a state-of-the-art deep learning implementation. In future work, we will incorporate to our presented system a fully fledged decoder and experiment on larger datasets, further investigating parameters of the visual feature extraction network.

6. References

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, 521:436–444, 2015.

[2] S. T. Shivappa, M. M. Trivedi, and B. D. Rao, "Audiovisual information fusion in human-computer interfaces and intelligent environments: A survey," *Proc. IEEE*, 98(10):1692–1715, 2010.

[3] G. Potamianos, E. Marcheret, Y. Mroueh, V. Goel, A. Koumbaroulis, A. Vartholomaios, and S. Thermos, "Audio and visual modality combination in speech processing applications," in *The Handbook of Multimodal-Multisensor Interfaces, Vol. 1*, S. Oviatt, B. Schuller, P. Cohen, D. Sonntag, G. Potamianos, and A. Krüger, Eds. Morgan Claypool, 2017.

[4] A. K. Katsaggelos, S. Bahaadini, and R. Molina, "Audiovisual fusion: Challenges and new approaches," *Proc. IEEE*, 103(9):1635–1653, 2015.

[5] Z. Zhou, G. Zhao, X. Hong, and M. Pietikäinen, "A review of recent advances in visual speech decoding," *Image Vision Comput.*, 32(9):590–605, 2014.

[6] E. Marcheret, G. Potamianos, J. Vopicka, and V. Goel, "Scattering vs. discrete cosine transform features in visual speech processing," in *Proc. FAAVSP*, pp. 175–180, 2015.

[7] G. Zhao, M. Barnard, and M. Pietikäinen, "Lipreading with local spatiotemporal descriptors," *IEEE Trans. Multimedia*, 11(7):1254–1265, 2009.

[8] M. Wand, J. Koutník, and J. Schmidhuber, "Lipreading with long short-term memory," in *Proc. ICASSP*, pp. 6115–6119, 2016.

[9] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng, "Multimodal deep learning," in *Proc. ICML*, pp. 689–696, 2011.

[10] Y. Mroueh, E. Marcheret, and V. Goel, "Deep multimodal learning for audio-visual speech recognition," in *Proc. ICASSP*, pp. 2130–2134, 2015.

[11] I. Almajai, S. Cox, R. Harvey, and Y. Lan, "Improved speaker independent lip reading using speaker adaptive training and deep neural networks," in *Proc. ICASSP*, pp. 2722–2726, 2016.

[12] K. Noda, Y. Yamaguchi, K. Nakadai, H. G. Okuno, and T. Ogata, "Audio-visual speech recognition using deep learning," *Appl. Intell.*, 42(4):722–737, 2015.

[13] Y. Li, Y. Takashima, T. Takiguchi, and Y. Ariki, "Lip reading using a dynamic feature of lip images and convolutional neural networks," in *Proc. ICIS*, pp. 1–6, 2016.

[14] T. Saitoh, Z. Zhou, G. Zhao, and M. Pietikäinen, "Concatenated frame image based CNN for visual speech recognition," in *Proc. ACCV Works.*, 2016.

[15] J. S. Chung, A. Senior, O. Vinyals, and A. Zisserman, "Lip reading sentences in the wild," *CoRR*, arXiv:1611.05358v2, 2017.

[16] Y. M. Assael, B. Shillingford, S. Whiteson, and N. de Freitas, "LipNet: End-to-end sentence-level lipreading," *CoRR*, arXiv:1611.01599v2, 2016.

[17] A. Graves and N. Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *Proc. ICML*, pp. 1764–1772, 2014.

[18] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak, "Convolutional, long short-term memory, fully connected deep neural networks," in *Proc. ICASSP*, pp. 4580–4584, 2015.

[19] W. Xiong, J. Droppo, X. Huang, F. Seide, M. Seltzer, A. Stolcke, D. Yu, and G. Zweig, "Achieving human parity in conversational speech recognition," *CoRR*, arXiv:1610.05256v2, 2017.

[20] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, 9(8):1735–1780, 1997.

[21] R. Jozefowicz, W. Zaremba, and I. Sutskever, "An empirical exploration of recurrent network architectures," in *Proc. ICML*, pp. 2342–2350, 2015.

[22] D. Neil, M. Pfeiffer, and S.-C. Liu, "Phased LSTM: Accelerating recurrent network training for long or event-based sequences," in *Adv. NIPS*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, Eds. pp. 3882–3890, 2016.

[23] A. Thanda and S. M. Venkatesan, "Audio visual speech recognition using deep recurrent neural networks," *CoRR*, arXiv:1611.02879, 2016.

[24] S. Petridis, Z. Li, and M. Pantic, "End-to-end visual speech recognition with LSTMs," in *Proc. ICASSP*, pp. 2592–2596, 2017.

[25] T. Stafylakis and G. Tzimiropoulos, "Combining residual networks with LSTMs for lipreading," *CoRR*, arXiv:1703.04105v2, 2017.

[26] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proc. ICML*, pp. 369–376, 2006.

[27] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Adv. NIPS*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. pp. 3104–3112, 2014.

[28] G. Potamianos and C. Neti, "Improved ROI and within frame discriminant features for lipreading," in *Proc. ICIP*, vol. 3, pp. 250–253, 2001.

[29] G. Potamianos and P. Scanlon, "Exploiting lower face symmetry in appearance-based automatic speechreading," in *Proc. AVSP*, pp. 79–84, 2005.

[30] G. Potamianos and C. Neti, "Audio-visual speech recognition in challenging environments," in *Proc. Eurospeech*, pp. 1293–1296, 2003.

[31] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *J. Mach. Learn. Res.*, 12:2121–2159, 2011.

[32] R. Girshick, "Fast R-CNN," in *Proc. ICCV*, pp. 1440–1448, 2015.

[33] Y. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, "Efficient BackProp," in *Neural Networks: Tricks of the Trade*, G. Montavon, G. B. Orr, and K.-R. Müller, Eds., pp. 9–48, vol. LNCS 7700, Springer, 2012,

[34] J. Tompson, R. Goroshin, A. Jain, Y. LeCun, and C. Bregler, "Efficient object localization using convolutional networks," in *Proc. CVPR*, pp. 648–656, 2015.

[35] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. ICML*, pp. 448–456, 2015.

[36] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, 15:1929–1958, 2014.

[37] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification," in *Proc. ICCV*, pp. 1026–1034, 2015.

[38] Y. Zhang, W. Chan, and N. Jaitly, "Very deep convolutional networks for end-to-end speech recognition," *CoRR*, arXiv:1610.03022, 2016.

[39] S. Chetlur, C. Woolley, P. Vandermersch, J. Cohen, J. Tran, B. Catanzaro, and E. Shelhamer, "cuDNN: Efficient primitives for deep learning," *CoRR*, arXiv:1410.0759v3, 2014.

[40] F. A. Gers, N. N. Schraudolph, and J. Schmidhuber, "Learning precise timing with LSTM recurrent networks," *J. Mach. Learn. Res.*, 3:115–143, 2003.

[41] R. Weide. The Carnegie Mellon pronouncing dictionary. [Online]. Available: http://www.speech.cs.cmu.edu/cgi-bin/cmudict

[42] J. Chorowski, D. Bahdanau, K. Cho, and Y. Bengio, "End-to-end continuous speech recognition using attention-based recurrent NN: First results," *CoRR*, arXiv:1412.1602, 2014.

[43] D. J. Kershaw, A. J. Robinson, and M. Hochberg, "Context-dependent classes in a hybrid recurrent network-HMM speech recognition system," in *Adv. NIPS*, D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, Eds., pp. 750–756, 1996.

[44] I. McGraw, R. Prabhavalkar, R. Alvarez, M. G. Arenas, K. Rao, D. Rybach, O. Alsharif, H. Sak, A. Gruenstein, F. Beaufays, and C. Parada, "Personalized speech recognition on mobile devices," in *Proc. ICASSP*, pp. 5955–5959, 2016.