Resource-efficient TDNN Architectures for Audio-visual Speech Recognition

Alexandros Koumparoulis
ECE Dept., Univ. of Thessaly
Volos, Greece
alkoumpa@uth.gr

Gerasimos Potamianos
ECE Dept., Univ. of Thessaly
Volos, Greece
gpotam@ieee.org

Samuel Thomas
IBM Research AI
Yorktown Heights, USA
sthomas@us.ibm.com

Edmilson da Silva Morais
IBM Research AI
São Paulo, Brazil
edmorais@br.ibm.com

***Abstract*—In this paper, we consider the problem of resource-efficient architectures for audio-visual automatic speech recognition (AVSR). Specifically, we complement our earlier work that introduced efficient convolutional neural networks (CNNs) for visual-only speech recognition, by focusing here on the sequence modeling component of the architecture, proposing a novel resource-efficient time-delay neural network (TDNN) that we extend for AVSR. In more detail, we introduce the sTDNN-F module, which combines the factored TDNN (TDNN-F) with grouped fully-connected layers and the shuffle operation. We then develop an AVSR system based on the sTDNN-F, incorporating the efficient CNNs of our earlier work and other standard visual processing and speech recognition modules. We evaluate our approach on the popular TCD-TIMIT corpus, under two speaker-independent training/testing scenarios. Our best sTDNN-F based AVSR system turns out 74% more efficient than a traditional TDNN one and 35% more efficient than TDNN-F, while maintaining similar recognition accuracy and noise robustness, and also significantly outperforming its audio-only counterpart.**

***Index Terms*—AVSR, TDNN, MobiLipNet, computational efficiency.**

I. Introduction

Recently, significant progress has been achieved in the area of audio-visual automatic speech recognition (AVSR), thanks to deep learning-based architectures, e.g. [1]–[14]. However, the resulting systems turn out to be computationally intensive, requiring significant hardware resources due to their reliance on expensive components, typically recurrent and convolutional neural networks (CNNs). For example, in the visual-only speech recognition (VSR) system of [14], the 2D-CNN alone includes 67.46×10^6 parameters and consumes 11.22×10^9 floating point operations (FLOPs) to process a single video frame. Such requirements render deployment on resource-limited devices impractical, thus necessitating the development of efficient AVSR architectures and motivating our work.

For traditional audio-only automatic speech recognition (ASR), resource-efficient deep-learning models have been investigated in many works. For example, in [15], efficiency was approached in a holistic manner, adapting multiple ASR pipeline components. Specifically, the parameters of the deep neural network-based acoustic model were quantized, allowing computations to utilize integer-only multiplication, while for decoding, lazy evaluation was used at the pre-softmax layer to reduce computational requirements by ignoring its denominator. Further, in other works, depthwise/pointwise convolution factorizations, initially proposed in efficient architectures for computer vision problems (e.g. MobileNet [16] for image classification), have been exploited in speech processing applications. For example, in [17], depthwise convolutions were used in a CNN-based system for audio-only single-channel speech separation, while for ASR such transformations have been applied to time-delay neural network (TDNN) architectures [18]–[22], gated CNN [23], and diagonal long short-term memory [24] based systems.

In contrast, the issues of computational cost and resource-efficient architectures have only recently started to attract interest in VSR and AVSR [25]–[29]. For example, in [25], computational efficiency of an AVSR system was evaluated by varying the recognition network's channel configuration. Further, in our early work on the topic [27], we introduced the "MobiLipNetV2" architecture for VSR, replacing standard convolutions of CNNs (used for visual stream encoding) by grouped ones, such as depthwise and pointwise, and utilizing the shuffle operation [30] to enable feature sharing across groups. As a result, we reduced computations by 37 times (in FLOPs) over the state-of-the-art 3D-ResNet with only a minor 0.07% absolute word error rate degradation for continuous VSR on TCD-TIMIT data [31]. Subsequently, in [28], we extended that work and proposed "MobiLipNetV3", coupled with a resource-adaptive scheme that employed two efficient CNNs to provide a range of operating points trading off VSR system efficiency vs. accuracy. Both our earlier papers relied on TDNNs for temporal classification, which are known to offer strong temporal modeling, while being leaner and easier to optimize compared to recurrent architectures. However, in those works we ignored the issue of efficient TDNN architectures, directing instead our attention to the more demanding CNN module. In contrast, in this paper, we shift our focus on improving the efficiency of TDNN-based modeling, thus complementing the efficient architectures of our earlier works [27], [28].

Specifically, as detailed in Section II, we introduce sTDNN-F, a novel resource-efficient module for TDNN-like architectures that is based on the factored TDNN [32], by replacing its fully-connected layers with grouped ones. Additionally, we insert a shuffle layer to the module, enabling feature sharing across parallel groups. It should be noted that the proposed sTDNN-F architecture is also applicable to audio-only recognizers, however our focus here lies primarily on AVSR. We thus build a resource-efficient AVSR system described in Section III, by integrating the proposed sTDNN-F with other standard visual processing and speech recognition components, in conjunction with early audio-visual fusion (feature concatenation) due to its popularity among AVSR works. We evaluate our approach on the TCD-TIMIT corpus [31], a widely used dataset for continuous speaker-independent AVSR, reporting our experiments in Section IV. There, we compare the proposed system to traditional TDNN baselines in terms of computational efficiency and recognition performance. Specifically, we consider a range of acoustic noise conditions for two system training/testing scenarios, and we demonstrate substantial efficiency gains by our approach with no significant degradation in recognition accuracy and noise robustness.

II. Network Modules

We now discuss our proposed sTDNN-F module, basing its presentation on the TDNN and TDNN-F modules that are frequently used in non-recurrent speech recognition systems, and examining the structure and computational cost of all three models in detail. Note

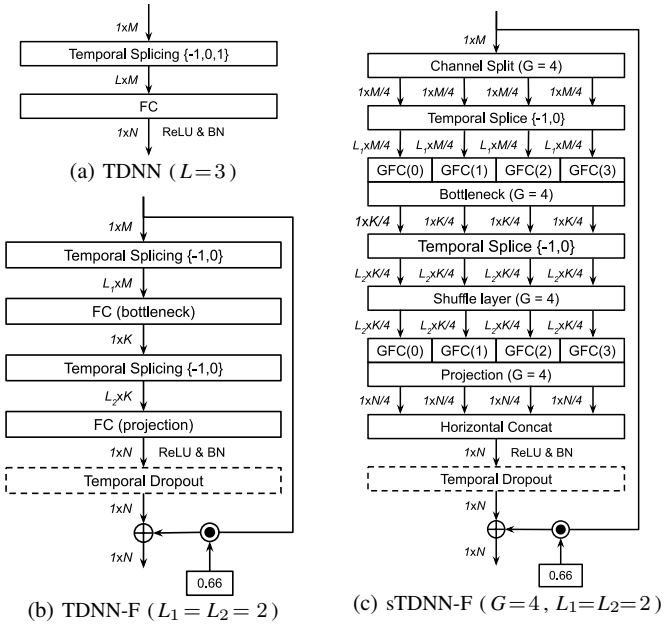

Fig. 1. The network building blocks detailed in Section II: (a) TDNN module with 3-frame splicing; (b) TDNN-F module with 2 FC layers and intermediate 2-frame splicing; (c) Proposed sTDNN-F module with 4 groups (sTDNN-F-4). Residual connections in the latter two are scaled by factor 0.66.

that in the following, we will be using the terms fully-connected (FC) and 1D-convolution interchangeably.

A. The TDNN module

TDNNs [18] (also known as 1D-CNNs) are built using FC layers of local temporal connectivity. A single module of such network is shown in Fig. 1a. We denote the dimension of the input feature vector with M and the output dimension with N. Initially, a total of L feature frames are concatenated to form a single feature vector ($1 \times M \rightarrow L \times M$). These are usually temporally adjacent for lower layers, or non-adjacent ones for layers deeper in the network (dilation greater than one in CNN terminology). The spliced feature vector is passed to an FC layer ($L \times M \rightarrow 1 \times N$). Finally, the rectified linear unit (ReLU) activation and batch normalization (BN) [33] are applied.

B. The TDNN-F module

The TDNN-F [32] is a factored form of TDNN. As also shown in Fig. 1b, it uses two (or more) FC layers, with one constrained to be semi-orthogonal. Similarly to a TDNN, the first layer is a splicing one that concatenates L_1 frames of M-dimensional features ($1 \times M \rightarrow L_1 \times M$). The first FC layer acts as bottleneck, by reducing the input dimension from M to K ($L_1 \times M \rightarrow 1 \times K$, with $K \ll M$). After that, L_2 frames are spliced again ($1 \times K \rightarrow L_2 \times K$), and the second FC layer (projection) is applied ($L_2 \times K \rightarrow 1 \times N$). There also exists a TDNN-F variant with an additional FC layer ($1 \times K \rightarrow 1 \times K$) between the bottleneck and projection layers, however we do not use it here. After the projection layer, ReLU and BN are applied. A special form of dropout is optionally used, where the mask is uniform across all time-steps and the dropout value has a continuous range instead of zero. A statically scaled (usually 0.66) residual connection is also applied between the input and output of the module.

C. The sTDNN-F module

The sTDNN-F module is a novel TDNN-F variant introduced here. Although the model retains the overall training procedure of its predecessor and its bottleneck and projection FC layers, the goal is to offer high recognition accuracy in a resource-efficient manner. For this purpose, the FC layers of the TDNN-F are replaced by so-called "grouped fully connected" (GFC) layers (see also Fig. 1c). In those layers, each output unit is only connected to a subset of the input. The modification yields clear efficiency improvements, due to reduced connectivity in FC layers. For example, if two groups were used, the required FLOPs would be halved. For notational convenience, we will append the number of groups employed to the naming convention of the proposed module, e.g. for a configuration that uses two groups we will refer to it as the sTDNN-F-2.

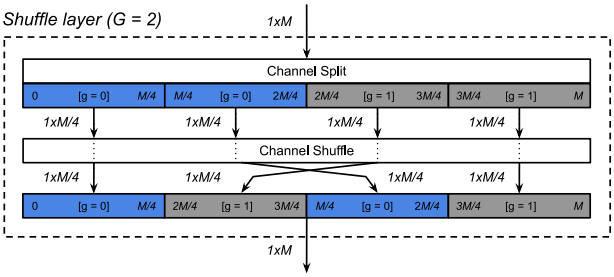

Fig. 2. Detailed operation of the sTDNN-F shuffle layer for M-dimensional features and $G = 2$ groups, depicting feature block re-positioning. Shown inside each block are the group id (e.g. $[g = 1]$), the start index (left side, inclusive), and the end index (right side, non-inclusive) of its elements.

An issue with parallel GFC layers is the lack of feature interchange across groups. To overcome this, we insert a shuffle layer [30], enabling cross-group information exchange. To highlight its operation, let us assume one input feature vector with two groups. Then, referring to Fig. 2, the first output group will contain the first-half features of both input groups, and similarly, the second output group will end up with the second-half features of the input groups. Note that the shuffle layer has no parameters and requires no computations: Memory copies can be eliminated by changing the destination location of the preceding layer, assuming that the shuffle layer is the sole successor, and thus there is no need for a feature vector in its original "non-shuffled" state.

In more detail, as shown in Fig. 1c, the sTDNN-F operation commences with a splitting layer that partitions the input feature vector to the number of groups (denoted by G). Next, for each group, a temporal splicing layer concatenates L_1 frames of M/G-dimensional features ($1 \times M/G \rightarrow L_1 \times M/G$). Then, a bottleneck GFC layer is applied on each group ($L_1 \times M/G \rightarrow 1 \times K/G$, with $K \ll M$). After the bottleneck layers, L_2 frames of dimension K/G are spliced again ($1 \times K/G \rightarrow L_2 \times K/G$), and the second set of GFC layers (projection) is applied ($L_2 \times K/G \rightarrow 1 \times N/G$). After the projection layer, ReLU and BN are employed. As in TDNN-F, dropout and residual connection are also used.

The factorization of sTDNN-F can be viewed as a within-module multi-stream TDNN-F variant with feature shuffling, where each group forms its own stream. Note that a multi-stream TDNN-F variant was presented in [34], where multiple parallel TDNN-F layers of different dilation were applied to the same input, without feature interchange between parallel streams.

D. Computational cost of the modules and comparison

We now consider the computational cost and size of the presented modules, while also discussing the efficiency gains achieved by the proposed sTDNN-F. Specifically:

- In the case of TDNN, its FC layer costs $2LMN + N$ FLOPs (in multiplications and additions), containing $LMN + N$ parameters (params), while BN costs $2N$ FLOPs and has $2N$ params.

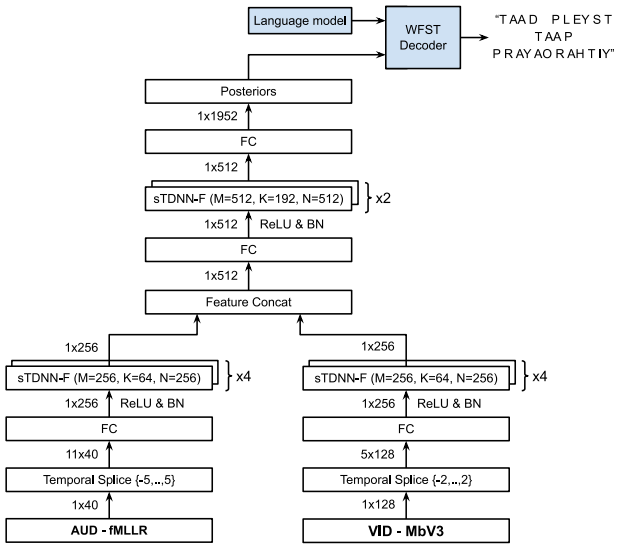

Fig. 3. The AVSR system of Section III, employing the proposed sTDNN-F modules, with their input (M), bottleneck (K), and output (N) dimensions listed. Depicted are the acoustic tower (*bottom left*) and the visual one (*bottom right*, further detailed in Fig. 4). Features from both are concatenated and fed to the bimodal network (*middle*). The network posteriors along with a bi-phone language model are sent to the weighted finite-state transducer (WFST) based decoder, yielding the recognized phonetic sequence (*top*).

- For the TDNN-F module, its first FC layer costs $2L_1MK + K$ FLOPs and has $L_1MK + K$ params, the second FC layer $2L_2KN + N$ FLOPs with $L_2KN + N$ params, and finally BN costs $2N$ FLOPs with $2N$ params.
- Finally, for the sTDNN-F, its bottleneck GFC layers require $2L_1(M/G)K+K$ FLOPs and $L_1(M/G)K+K$ params, the projection GFC ones cost $2L_2(K/G)N+N$ FLOPs and $L_2(K/G)N+N$ params, while BN consumes $2N$ FLOPs with $2N$ params.

To provide an efficiency comparison between the three modules, we assume identical input and output dimensions ($M=N$). Further, for TDNN-F and sTDNN-F, we assume bottleneck dimensions equal to a quarter of the input ones ($K = M/4$). Finally, we assume that the TDNN employs symmetric splicing of the current, $\lfloor L/2 \rfloor$ past, and $\lfloor L/2 \rfloor$ future frames, with odd $L = 3$, and that in the TDNN-F and sTDNN-F modules splicing occurs with $L_1 = L_2 = 2$ (current and one past frame), as in our experiments.

Under these assumptions, the TDNN module requires $6M^2 + M$ FLOPs, the TDNN-F costs $2M^2 + 3M/4$ FLOPs, whereas for our proposed module, the cost becomes $M^2 + 5M/4$ for sTDNN-F-2 and $M^2/2 + 5M/4$ for sTDNN-F-4. If we approximate these costs using only the squared terms, then TDNN-F requires 66% less FLOPs than TDNN, while sTDNN-F-2 and sTDNN-F-4 require 83% and 92% less FLOPs respectively, regardless of the value of M. It can be easily deduced that similar gains hold for the model size (params).

III. Recognition Systems

Next, we utilize the aforementioned modules in the AVSR system architecture, together with suitable visual processing and speech recognition components. In addition, we build an audio-only ASR counterpart for performance comparisons. For development we exploit the Kaldi framework [35], and we employ weighted finite-state transducer (WFST) based decoding that incorporates a bi-phone language model (suitable for the recognition task of Section IV).

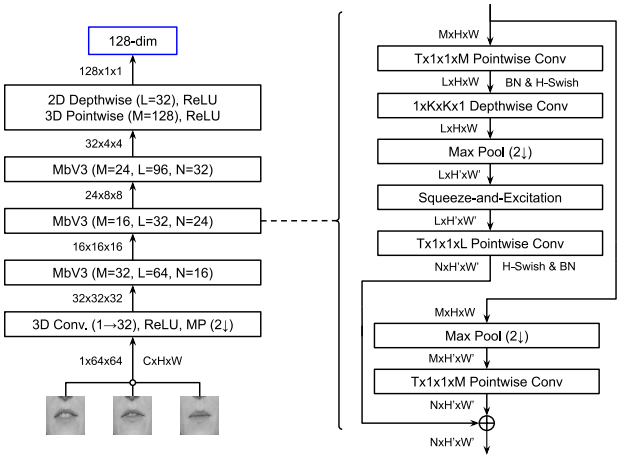

Fig. 4. *Left:* The five-layer 3D-pointwise MobiLipNetV3 (MbV3) CNN, introduced in our earlier work [28] for efficient visual stream representation and employed here for AVSR (visual tower of Fig. 3). Its input is formed by splicing three consecutive mouth regions (see also Fig. 5). The input and output dimensions of each CNN layer are also shown (in the format: number of channels $C \times$ height $H \times$ width W). *Right:* Detailed layer architecture of the MbV3 module. More information can be found in [28].

A. ASR system

Our ASR system is bootstrapped from a traditional GMM-HMM with MFCC features. This yields frame targets via forced alignment, which help train a TDNN-WFST hybrid system on 40-dimensional fMLLR features, whose architecture is similar to the left sub-network of our AVSR system (Fig. 3). In more detail, a splicing layer concatenates 11 fMLLR feature frames (current, 5 past, and 5 future ones) that are then processed by an FC layer, followed by ReLU and BN. Next, five sTDNN-F modules are applied, and a projection layer maps the resulting representation to 1952 context-dependent tied-triphone HMM states. Two ASR implementations are considered, one with sTDNN-F modules of input dimensionality $M = 256$ and a larger one with $M = 1024$. In both cases, $M = N$ and $K = M/4$.

B. AVSR system

Our AVSR system, shown in Fig. 3, consists of one sub-network for each modality and an audio-visual one that receives features from both. The audio part is identical to the ASR network described above, up to the final projection layer that is removed. On the other hand, the visual sub-network (Fig. 3, right) commences with the MobiLipNetV3 model, i.e. the resource-efficient five-layer CNN of our recent work [28], also depicted in Fig. 4 for easy reference. The model operates on three consecutive grayscale mouth regions, extracted as described later, and it outputs 128-dimensional visual representations at the video frame rate (30 Hz). These are upsampled linearly to match the audio feature rate (100 Hz) and are further processed quite similarly to the audio features. Specifically, five visual feature frames are spliced together, an FC layer is applied that is followed by ReLU and BN, and last, four sTDNN-F modules are employed. At the final part of the AVSR system architecture, the unimodal representations of the two sub-networks are concatenated (early fusion). The resulting features are processed by the bimodal network that consists of an FC layer that is followed by ReLU and BN, two sTDNN-F modules, and a projection layer that yields HMM state posteriors. Note that all unimodal sTDNN-F modules in our AVSR system have $M = N = 256$ and $K = 64$, while the bimodal ones use $M = N = 512$ and $K = 192$.

For speaker mouth localization and region-of-interest (ROI) extraction, the pipeline of [27] is used, as also depicted in Fig. 5. First, the

Fig. 5. The visual pre-processing pipeline for extracting the mouth region-of-interest (ROI) that is then fed to the CNN of Fig. 4.

input frame is converted to grayscale, and face detection is performed using a ResNet-10 with single-shot detector network [36], available in OpenCV v3.4 [37]. Then, facial landmarks are detected as in [38]. From those, four mouth landmarks are used, after median filtering over a 7-frame window, to yield smooth mouth center, width, and height estimates. Based on these, a grayscale mouth ROI is extracted (approximately enlarged by 40% over the mouth width and height), normalized to 64×64 pixels, to be fed to the MobiLipNetV3 CNN.

IV. Experiments

A. Dataset and experimental framework

Our experiments are run on the popular TCD-TIMIT corpus [31]. This is about 8 hours in total duration, containing continuous speech by 59 subjects that utter phonetically-rich TIMIT sentences with a medium-size 6k-word vocabulary, recorded in ideal, studio-like conditions by two cameras and a lapel microphone.

In the experiments, the frontal-view videos of the database are used, available at a 1920×1080-pixel resolution and 30 Hz video frame rate. Further, to simulate noisy conditions, the clean audio data of the corpus are distorted by additive white Gaussian noise at six signal-to-noise (SNR) levels ranging from 20 dB to -5 dB, thus allowing for various training/testing scenarios to be considered. Here, two ASR and two AVSR models are built for each of the TDNN-based architectures investigated: (i) models trained on clean audio data, thus evaluated on mismatched/unseen noise conditions (with a notable exception when tested on clean audio), and (ii) models trained on data from all audio conditions combined (referred to as multi-condition training). Note that all recognition results are reported on the official corpus speaker-independent test set (17 subjects) with 39 subjects used for training and 3 for validation, thus differing from the setup of [39] where two test subsets were considered.

The evaluation of all ASR and AVSR systems employing different TDNN-based modules concerns both their recognition performance and resource efficiency. The former is reported on the TCD-TIMIT test set in terms of phone error rate (PER), due to the data nature (TIMIT recognition task), while the latter in terms of required computations (in FLOPs per input frame) and number of parameters used in all TDNN-based modules of the considered architectures.

B. Results

Our experimental results are reported in Table I, concerning four different variations of TDNN-based modules, namely the traditional TDNN (baseline), its factored variant (TDNN-F), and two versions

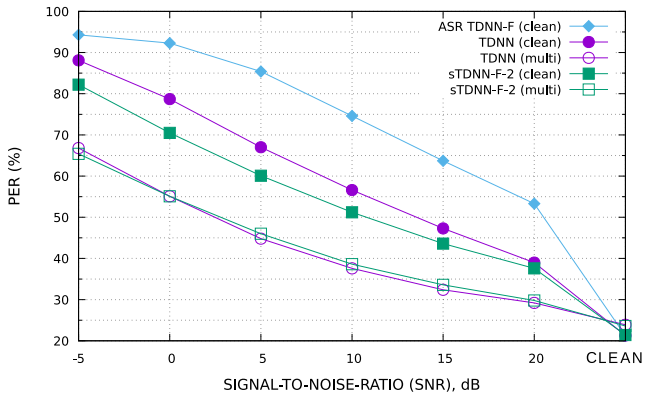

Fig. 6. Recognition performance on the TCD-TIMIT test set (in PER, %) of various AVSR systems of Table I, as well as of the 1024-channel TDNN-F audio-only ASR system (training conditions specified inside parenthesis).

of the proposed (sTDNN-F) with 2 and 4 groups. The table primarily refers to the AVSR systems, with efficiency reported at its left-most part and performance at various SNRs in its middle part for models trained on clean audio or on all conditions. Note that audio-only ASR performance on clean conditions is provided at the right-most section.

Concerning the AVSR models of Table I, it can be observed that the most efficient architecture is the sTDNN-F-4, having for example 81% less FLOPs and 67% less parameters than the TDNN baseline. However, it yields the worst PERs among all systems under multi-condition training. In contrast, the proposed sTDNN-F-2 module offers a better efficiency-performance compromise, remaining significantly more computationally efficient than both the TDNN (by 74%) and TDNN-F (by 35%, i.e. 2.48M vs. 3.79M FLOPs per input frame), while also generalizing well in the mismatched scenario (clean-audio training). There, it outperforms the more expensive TDNN across all SNRs (except in clean conditions) and lags slightly behind TDNN-F by an average of 0.65% absolute PER degradation over all SNRs. Further, under multi-condition training, it outperforms the TDNN-F at most SNRs, yet slightly lags the best-performing but expensive TDNN by at most 1.4% in absolute PER degradation.

Some of the above AVSR results are also depicted in Fig. 6 for better visualization. It is evident that the sTDNN-F-2 outperforms the TDNN baseline when both are trained on clean audio, but slightly lags it under multi-condition training. In addition to AVSR plots, also shown is the performance of the best audio-only system (TDNN-F with $M = 1024$) when trained on clean audio. It is evident that audio-only ASR degrades rapidly as the SNR decays, significantly trailing AVSR performance that exhibits better noise robustness.

The right-most section of Table I is dedicated to the performance of audio-only ASR systems, trained and tested on clean conditions. As mentioned in Section III-A, two module configurations are considered ($M = 256$ and 1024) for all architectures. Note first that in all cases, the audio-only PERs trail those of the corresponding AVSR systems

TABLE I

AVSR evaluation of four TDNN variations. Left-to-right: Efficiency in per-frame FLOPs and parameters (in millions, also shown in % relative to baseline); Performance in PER (%) on the TCD-TIMIT test set at various SNRs when models are trained on clean audio or on all noise conditions jointly; PER of audio-only ASR models trained and tested on clean audio for $M = 256$ and 1024.

Network	FLOPs		parameters		PER (dB) – clean-audio training							PER (dB) – multi-condition training							PER (ASR)	
	(M)	rel.(%)	(M)	rel.(%)	clean	20	15	10	5	0	-5	clean	20	15	10	5	0	-5	256	1024
TDNN	9.56	–	5.78	–	21.3	39.0	47.3	56.6	67.0	78.7	88.1	23.9	29.2	32.4	37.6	44.8	55.1	66.8	22.5	21.7
TDNN-F	3.79	60.35	2.89	50.00	20.7	37.5	43.7	50.7	58.7	69.6	81.3	23.7	29.9	33.5	38.9	46.7	56.1	65.6	22.8	21.2
sTDNN-F-2	2.48	74.05	2.24	61.24	21.4	37.6	43.6	51.2	60.1	70.5	82.2	23.6	29.8	33.6	38.6	46.0	55.5	65.4	24.1	22.0
sTDNN-F-4	1.82	80.96	1.91	66.95	21.5	37.2	43.4	51.5	60.8	71.6	82.8	24.9	31.0	34.4	39.4	47.0	56.8	67.1	25.0	22.3

(trained and tested on clean audio). Note also that the lowest audio-only PER is obtained by the 1024-channel TDNN-F (21.2%), followed by the 1024-channel TDNN (21.7%). The proposed sTDNN-F networks suffer significant PER degradation for $M = 256$, reaching up to 25.0% PER. However, for $M = 1024$, they become competitive: For example, compared to TDNN-F, the sTDNN-F-2 system yields a slight only PER degradation (from 21.2% to 22.0% PER), but at the benefit of significantly superior efficiency. Indeed, although not shown in Table I, the sTDNN-F-2 requires significantly less FLOPs (3.92M vs. 11.78M of the TDNN-F, per input frame), as well as number of module parameters (3.76M vs. 7.70M). Finally, note that the reported PERs are competitive to the literature, for example [39] reports 23.5% and 21.6% PER on two test subsets of TCD-TIMIT, using a six FC-layer DNN with 11-frame input splicing.

V. Conclusions

In this paper, we proposed a novel module for TDNN-like AVSR network architectures, by replacing the fully-connected TDNN-F layers with grouped ones for improved efficiency and inserting a shuffle layer for retaining recognition performance. We integrated the proposed module to an AVSR system, together with other visual processing and speech recognition components, and we investigated its performance and efficiency compared to TDNN-based alternatives on the TCD-TIMIT corpus under two speaker-independent training/testing scenarios. The resulting sTDNN-F-2 based AVSR system turned out 74% and 35% more efficient than the TDNN and the TDNN-F ones, respectively, while maintaining similar recognition accuracy and noise robustness.

References

[1] G. Potamianos *et al.*, "Audio and visual modality combination in speech processing applications," in *The Handbook of Multimodal-Multisensor Interfaces, Vol. 1*, S. Oviatt *et al.* (Eds.) Morgan-Claypool, 2017, pp. 489–543.

[2] T. Afouras, J. S. Chung, A. Senior, O. Vinyals, and A. Zisserman, "Deep audio-visual speech recognition," *IEEE Trans. Pattern Anal. Machine Intell. (Early Access)*, 2018.

[3] S. Petridis, T. Stafylakis, P. Ma, G. Tzimiropoulos, and M. Pantic, "Audio-visual speech recognition with a hybrid CTC/attention architecture," in *Proc. SLT*, 2018, pp. 513–520.

[4] T. Makino, H. Liao, Y. Assael, B. Shillingford, B. Garcia, O. Braga, and O. Siohan, "Recurrent neural network transducer for audio-visual speech recognition," in *Proc. ASRU*, 2019, pp. 905–912.

[5] J. Yu, S. Zhang, J. Wu, S. Ghorbani, B. Wu, S. Kang, S. Liu, X. Liu, H. Meng, and D. Yu, "Audio-visual recognition of overlapped speech for the LRS2 dataset," in *Proc. ICASSP*, 2020, pp. 6984–6988.

[6] S. Zhang, M. Lei, B. Ma, and L. Xie, "Robust audio-visual speech recognition using bimodal DFSMN with multi-condition training and dropout regularization," in *Proc. ICASSP*, 2019, pp. 6570–6574.

[7] P. Zhou, W. Yang, W. Chen, Y. Wang, and J. Jia, "Modality attention for end-to-end audio-visual speech recognition," in *Proc. ICASSP*, 2019, pp. 6565–6569.

[8] G. Sterpu, C. Saam, and N. Harte, "Attention-based audio-visual fusion for robust automatic speech recognition," in *Proc. ICMI*, 2018, pp. 111–115.

[9] G. Sterpu, C. Saam, and N. Harte, "How to teach DNNs to pay attention to the visual modality in speech recognition," *IEEE/ACM Trans. Audio Speech Language Process.*, 28:1052–1064, 2020.

[10] F. Tao and C. Busso, "Gating neural network for large vocabulary audio-visual speech recognition," *IEEE/ACM Trans. Audio Speech Language Process.*, 26(7):1290–1302, 2018.

[11] M. Wand, J. Schmidhuber, and N. Vu, "Investigations on end-to-end audiovisual fusion," in *Proc. ICASSP*, 2018, pp. 3041–3045.

[12] S. Petridis, T. Stafylakis, P. Ma, F. Cai, G. Tzimiropoulos, and M. Pantic, "End-to-end audiovisual speech recognition," in *Proc. ICASSP*, 2018, pp. 6548–6552.

[13] S. Petridis, Y. Wang, Z. Li, and M. Pantic, "End-to-end audiovisual fusion with LSTMs," in *Proc. AVSP*, 2017, pp. 36–40.

[14] J. S. Chung, A. Senior, O. Vinyals, and A. Zisserman, "Lip reading sentences in the wild," in *Proc. CVPR*, 2017, pp. 3444–3453.

[15] V. Vanhoucke, A. Senior, and M. Z. Mao, "Improving the speed of neural networks on CPUs," in *Proc. Deep Learning and Unsupervised Feature Learning NIPS Works.*, 2011.

[16] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," *CoRR*, abs/1704.04861, 2017.

[17] Y. Luo and N. Mesgarani, "Conv-TasNet: Surpassing ideal time-frequency magnitude masking for speech separation," *IEEE/ACM Trans. Audio Speech Language Process.*, 27(8):1256–1266, 2019.

[18] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. J. Lang, "Phoneme recognition using time-delay neural networks," *IEEE Trans. Acoustics Speech Signal Process.*, 37(3):328–339, 1989.

[19] S. Kriman, S. Beliaev, B. Ginsburg, J. Huang, O. Kuchaiev, V. Lavrukhin, R. Leary, J. Li, and Y. Zhang, "QuartzNet: deep automatic speech recognition with 1D time-channel separable convolutions," in *Proc. ICASSP*, 2020, pp. 6124–6128.

[20] A. Hannun, A. Lee, Q. Xu, and R. Collobert, "Sequence-to-Sequence speech recognition with time-depth separable convolutions," in *Proc. Interspeech*, 2019, pp. 3785–3789.

[21] V. Pratap, Q. Xu, J. Kahn, G. Avidov, T. Likhomanenko, A. Hannun, V. Liptchinsky, G. Synnaeve, and R. Collobert, "Scaling up online speech recognition using ConvNets," in *Proc. Interspeech*, 2020, pp. 3376–3380.

[22] Y. Fujita, A. S. Subramanian, M. Omachi, and S. Watanabe, "Attention-based ASR with lightweight and dynamic convolutions," in *Proc. ICASSP*, 2020, pp. 7034–7038.

[23] J. Park, X. Qian, Y. Jo, and W. Sung, "Low-latency lightweight streaming speech recognition with 8-bit quantized simple gated convolutional neural networks," in *Proc. ICASSP*, 2020, pp. 1803–1807.

[24] W. Sung, L. Lee, and J. Park, "Exploration of on-device end-to-end acoustic modeling with neural networks," in *Proc. SiPS*, 2019, pp. 160–165.

[25] M. Van keirsbilck, B. Moons, and M. Verhelst, "Resource aware design of a deep convolutional-recurrent neural network for speech recognition through audio-visual sensor fusion," *CoRR*, abs/1803.04840, 2018.

[26] N. Shrivastava, A. Saxena, Y. Kumar, R. R. Shah, A. Stent, D. Mahata, P. Kaur, and R. Zimmermann, "MobiVSR: Efficient and light-weight neural network for visual speech recognition on mobile devices," in *Proc. Interspeech*, 2019, pp. 2753–2757.

[27] A. Koumparoulis and G. Potamianos, "MobiLipNet: Resource-efficient deep learning based lipreading," in *Proc. Interspeech*, 2019, pp. 2763–2767.

[28] A. Koumparoulis, G. Potamianos, S. Thomas, and E. S. Morais, "Resource-adaptive deep learning for visual speech recognition," in *Proc. Interspeech*, 2020, pp. 3510–3514.

[29] P. Ma, B. Martinez, S. Petridis, and M. Pantic, "Towards practical lipreading with distilled and efficient models," in *Proc. ICASSP*, 2021, pp. 7608–7612.

[30] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: an extremely efficient convolutional neural network for mobile devices," in *Proc. CVPR*, 2018, pp. 6848–6856.

[31] N. Harte and E. Gillen, "TCD-TIMIT: An audio-visual corpus of continuous speech," *IEEE Trans. Multimedia*, 17(5):603–615, 2015.

[32] D. Povey, G. Cheng, Y. Wang, K. Li, H. Xu, M. Yarmohammadi, and S. Khudanpur, "Semi-orthogonal low-rank matrix factorization for deep neural networks," in *Proc. Interspeech*, 2018, pp. 3743–3747.

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

[34] K. Han, J. Pan, V. Tadala, T. Ma, and D. Povey, "Multistream CNN for robust acoustic modeling," *CoRR*, abs/2005.10470, 2020.

[35] D. Povey *et al.*, "The Kaldi speech recognition toolkit," in *Proc. ASRU*, 2011.

[36] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," in *Proc. ECCV*, 2016, pp. 21–37.

[37] G. Bradski and A. Kaehler, *Learning OpenCV: Computer Vision with the OpenCV Library*. O'Reilly Media, 2008.

[38] S. Ren, X. Cao, Y. Wei, and J. Sun, "Face alignment at 3000 fps via regressing local binary features," in *Proc. CVPR*, 2014, pp. 1685–1692.

[39] A. H. Abdelaziz, "NTCD-TIMIT: A new database and baseline for noise-robust audio-visual speech recognition," in *Proc. Interspeech*, 2017, pp. 3752–3756.