Neural Speaker Embeddings for Ultrasound-based Silent Speech Interfaces

Amin Honarmandi Shandiz[1], *László Tóth*[1], *Gábor Gosztolya*[2], *Alexandra Markó*[3,4],
Tamás Gábor Csapó[4,5]

[1]Institute of Informatics, University of Szeged, Hungary
[2]MTA-SZTE Research Group on Artificial Intelligence, Szeged, Hungary
[3]Department of Applied Linguistics and Phonetics, Eötvös Loránd University, Budapest, Hungary
[4]MTA-ELTE Lendület Lingual Articulation Research Group, Budapest, Hungary
[5]Department of Telecommunications and Media Informatics,
Budapest University of Technology and Economics, Budapest, Hungary

{shandiz, tothl, ggabor}@inf.u-szeged.hu, marko.alexandra@btk.elte.hu, csapot@tmit.bme.hu

Abstract

Articulatory-to-acoustic mapping seeks to reconstruct speech from a recording of the articulatory movements, for example, an ultrasound video. Just like speech signals, these recordings represent not only the linguistic content, but are also highly specific to the actual speaker. Hence, due to the lack of multi-speaker data sets, researchers have so far concentrated on speaker-dependent modeling. Here, we present multi-speaker experiments using the recently published TaL80 corpus. To model speaker characteristics, we adjusted the x-vector framework popular in speech processing to operate with ultrasound tongue videos. Next, we performed speaker recognition experiments using 50 speakers from the corpus. Then, we created speaker embedding vectors and evaluated them on the remaining speakers. Finally, we examined how the embedding vector influences the accuracy of our ultrasound-to-speech conversion network in a multi-speaker scenario. In the experiments we attained speaker recognition error rates below 3%, and we also found that the embedding vectors generalize nicely to unseen speakers. Our first attempt to apply them in a multi-speaker silent speech framework brought about a marginal reduction in the error rate of the spectral estimation step.

Index Terms: silent speech interface, articulatory-to-acoustic mapping, speaker embedding, x-vector

1. Introduction

Silent Speech Interfaces (SSI) aim to convert silent (mouthed) articulation to audible speech [1, 2, 3]. Such SSI systems could aid the communication of the speaking impaired (e.g. patients after laryngectomy). The theoretical background is provided by articulatory-to-acoustic mapping (AAM), where articulatory data is recorded while the subject is speaking, and machine learning methods (typically deep neural networks (DNNs)) are applied to predict the speech signal from the articulatory input. The set of articulatory acquisition devices includes ultrasound tongue imaging (UTI) [4, 5, 6, 7, 8], Magnetic Resonance Imaging (MRI) [9], electromagnetic articulography (EMA) [10, 11, 12], permanent magnetic articulography (PMA) [13, 14, 15], surface electromyography (sEMG) [16, 17, 18], electro-optical stomatography (EOS) [19], lip videos [20, 21], or a multimodal combination of the above [22].

Although there are lots of studies on generating intelligible speech from the above biosignals, most of these were conducted on relatively small databases of just a single or a small number of speakers [1, 2, 3]. Meanwhile, all of the articulatory tracking devices are obviously highly sensitive to the actual speaker's anatomy. A further source of variance may come from the possible misalignment of the recording equipment. For example, for ultrasound recordings, the probe fixing headset has to be remounted onto the speaker for each recording session. This inevitably causes the recorded ultrasound videos to become slightly misaligned between each recording session.

Although the properly working cross-session and cross-speaker methodologies are still missing, there have already been studies in this direction. Kim et al. investigated speaker-independent Silent Speech Recognition (SSR) using EMA with 12 healthy and laryngectomized speakers [11]. For EMG-based recognition, several signal normalization and model adaptation methods were investigated by Maier-Hein et al. [16]. Janke et al. studied session-independent sEMG over 16 sessions of a speaker, and the results showed that sEMG is quite robust to minor changes in the electrode placement [17]. Wand et al. utilized domain-adversarial DNN training to increase the session-independency of their EMG-based speech recognizer [18]. For EOS-based SSR with a small vocabulary, Stone and Birkholz found the speaker-independent average word accuracy to be relatively stable, varying between 56–62% [19].

Ultrasound-based SSI systems, however, might be less robust, as slight changes in probe positioning causes shifts and rotations in the resulting image. To this end, we examined on the session dependency of UTI-based direct speech synthesis, and we proposed a simple session adaptation method [23]. Ribeiro et al. experimented with the classification of UTI images of phonetic segments [24], and found that speaker-dependent systems perform much better than speaker-independent ones, but adding speaker information in a simple form (e.g. mean ultrasound image) helps the model generalize to unseen speakers. The same authors reported that unsupervised model adaptation can improve the results for silent speech (but not for modal speech) [6]. They also performed multi-speaker recognition and synthesis experiments where they applied x-vectors for speaker conditioning – but they extracted the x-vectors from the acoustic data and not from the ultrasound [25].

1.1. Speaker embedding vectors

Just like the above-mentioned biosignals, speech signals also contain speaker-specific factors that hurt the cross-speaker performance of speech processing systems. Various DNN training

and adaptation methods have been proposed as remedies, perhaps the simplest being is to use auxiliary input features that encapsulate information on speaker characteristics. The classic such representation was the i-vector [26], but its role has recently been taken by the x-vector [27]. The x-vector can be created by training a DNN for speaker classification, and then the activation values of an upper hidden layer are used as the speaker embedding vector in other tasks. Our goal here is to adjust this x-vector scheme to ultrasound tongue videos. To this aim, we train a DNN for speaker classification, using 3D blocks of adjacent ultrasound frames as input. First, we report the speaker recognition accuracy of this network in 5.1. Next, we evaluate the speaker embedding vector provided by this network on a separate set of speakers via very simple speaker recognition experiments in 5.2. Finally, we attempt to apply the embedding vectors as auxiliary input for a second network which is trained to perform speech synthesis, more precisely, to estimate spectral vectors from the ultrasound frames (5.3). Before the experiments, brief descriptions are given regarding our SSI framework, x-vectors and the experimental conditions in Sections 2, 3 and 4, respectively.

2. The SSI framework

We presented our approach for creating an ultrasound-based SSI in [5], so we just give a brief overview here. As Fig 1 shows, the input to our system is a sequence of ultrasound tongue imaging (UTI) frames, and the target sequence is a speech signal. This is a sequence-to-sequence mapping problem, which could be addressed by sophisticated encoder-decoder networks that would not even require aligned training data [25]. However, as we have synchronized input-output samples, most authors apply simpler networks that perform the mapping frame by frame [28, 5]. The optimal output representation is also a matter of choice. While training DNNs that generate speech signals directly is feasible [29], it would require large amounts of training data. Alternatively, one can use a dense spectral representation as the training target that can be converted to speech [4]. Recently, speech synthesis technology has introduced neural vocoders for synthesizing speech from spectrograms [29]. The main advantage of applying these in the SSI task is that that we can borrow large pre-trained networks for the speech synthesis step, and we have to deal only with the ultrasound-to-spectrum mapping task. In a recent study we used WaveGlow [5], and we obtained higher quality speech than with standard vocoders [4].

As WaveGlow requires a mel-spectrogram with 80 spectral components as input, our DNN was trained to estimate such spectral vectors from the UTI video in a frame-by-frame manner. The input of our network is a 3D array of consecutive images, and the output is a 80-dimensional spectral vector. The convolutional (3D-CNN) network structure that we applied here [30] was the same as the lower, 'frame-level' part of the x-vector network, so we delay its presentation to the next section. As here the task is to estimate spectral vectors, we used a linear output layer and the network was trained to minimize the mean-squared error (MSE) of the regression task.

3. Speaker embedding vectors for ultrasound tongue imaging

The x-vector concept was introduced by Snyder et al., motivated by the goal of replacing the previous Gaussian-based i-vector approach with a purely neural solution [27]. The basis behind the x-vector is a DNN that is trained to discriminate
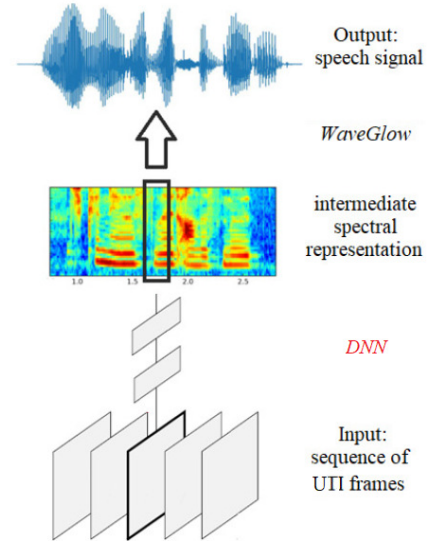

Figure 1: *Schematic diagram of the UTI-to-speech conversion process applied in our SSI model.*

speakers. The network structure is unusual in the sense that it consists of three main parts. The lower layers – typically a time-delay network (TDNN [31]) – operate on the level of frames. Then, the subsequent temporal pooling layer aggregates statistics over the frames of a given speech segment or utterance. This layer may collect only the mean values [32], the mean and the standard deviation [27, 33, 34] or it may even apply a sophisticated attention mechanism [35, 36]. The aggregated values are processed further by several fully connected layers. As these layer operate on the segment level, after training they can produce a fixed-size speaker embedding vector even from utterances of variable lengths. The embedding vector is typically obtained as the linear activation output of the fully connected layer right below or two layers below the softmax output [27].

We adjusted the x-vector DNN to operate with a sequence of ultrasound images instead of a sequence of speech feature vectors as follows (see also Fig 2).

The **frame-level part** has the same structure as the network we apply in the spectral estimation step of Fig 1 [30]. Its input consists of 21 consecutive images, which are processed by a 3D convolutional layer in 5-image blocks, using a relatively large stride size of 4 along the time axis. The subsequent convolutional layers apply the 3D convolution in a decomposed "(2+1)D" form, first processing these blocks locally, and aggregating their content along time only at higher layers. This architecture was motivated by the findings of Tran et al. for video recognition [37], and also by the success of TDNNs in speech recognition [31, 38]. The top layer of the frame-level part is a dense layer that produces a local output vector at each frame position (ie., for the center frame of the input block).

The **statistical pooling** layer performs a simple average pooling. We leave more complex solutions such as standard deviation pooling or attention-based weighting for future work.

The **segment-level part** consists of two fully connected layers and a softmax output layer that has one output neuron for each speaker in the training set. The neural speaker embedding vector for a given input segment or utterance is extracted from one of these dense layers (see the experiments in Section 5).
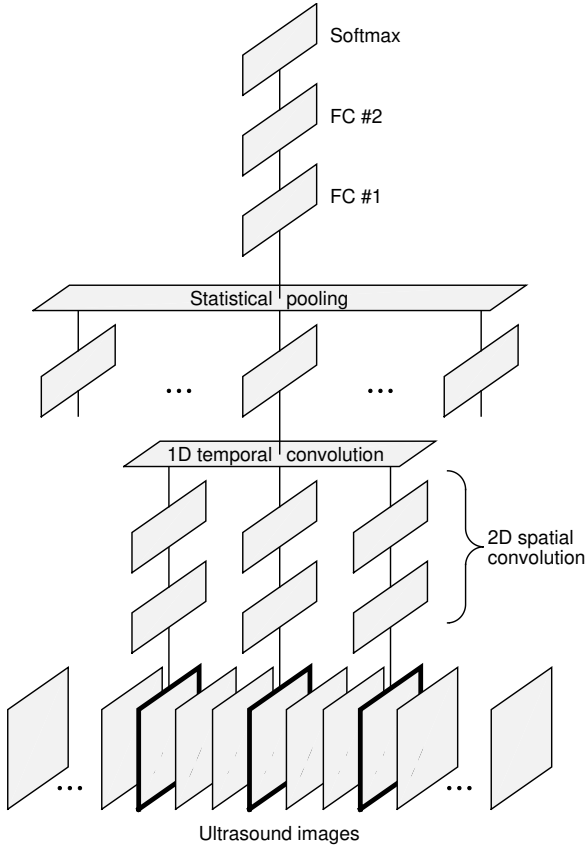
Figure 2: *Illustration of the UTI-based x-vector network.*

4. Experimental Set-Up

In the experiments we used the TaL80 corpus [25], which contains ultrasound, speech and lip video recordings from 81 speakers. Apart from the silent speech experiments, the speech signals were also recorded in parallel with the ultrasound, and here we used these synchronized ultrasound and speech tracks. The ultrasound was recorded using Articulate Instruments' Micro system that captures a midsaggital view of the tongue at a frame rate of 82 fps. The raw ultrasound images contain 64 x 842 data values, which were resized to 64 x 128 pixels. More details about the recording process can be found in [25].

We divided the data into two sets, so we used 50 speakers to create the x-vector network, and we held out 31 speakers to train and evaluate the SSI network. Although the x-vector network is able to handle inputs with different sizes, for technical simplicity we chose to work with short uniform 2-second long chunks from each recording (similar to Shon et al. [34]). To train the x-vector network, we extracted such 2-second (164 frames) chunks from each training file of the 50-speaker subset of the corpus, resulting in 76 chunks from each speaker on the average (minimum 65, maximum 85). Altogether we obtained 3800 such blocks, from which 2298 were used for training, 357 for development, and 1145 blocks for testing.

The network used for the spectral estimation task in the SSI and the frame-level part of the x-vector network were structurally identical, consisting of 4 3D-convolutional layers (with a MaxPooling layer after every second convolution), and a fully connected layer that aggregates the convolutional outputs (the exact model parameters can be found in our earlier work [30]).

Table 1: *Speaker recognition error rates for the 50-speaker set as a function of the input segment duration.*

Segment length (frames & seconds)	Aggregated frames	Error rate (Dev Set)
21 (0.25 sec)	1	3.16%
41 (0.50 sec)	21	2.87%
82 (1.0 sec)	62	2.10%
164 (2.0 sec)	144	1.96%

The fully connected hidden layer was followed by a linear output layer for the spectral estimation task, while in the x-vector network the fully connected layer outputs were aggregated along time by average pooling. The segment-level part consisted of two fully connected layers (FC#1 and FC#2 in Fig 2) with sizes of 500 and 250, and the softmax output layer contained 50 neurons, corresponding to the 50 speakers of the train set. Both networks were trained using the Adam optimizer with a learning rate of 0.0002, using a batch size of 100 for the SSI network, and batch sizes of 4-16 for the x-vector network (using smaller batch sizes for longer segments). The nonlinearity applied in all hidden layers was the swish function.

5. Results and Discussion

5.1. Training the x-vector network

In the first experiment we were interested to measure the speaker classification accuracy of our x-vector network, and how it is influenced by the input duration. The motivation for the aggregation step in the original x-vector model is that normally we have not just a single frame but whole utterances (at least a word) from a given speaker, and that certain phonetic segments may be less speaker-discriminative than others [34] – the most trivial example are the silent parts. But the situation is different when the input is an ultrasound tongue video, as the recording device always returns an image of the tongue, even when the speaker remains silent, and thus longer segments may not be necessary. In the first experiment we gradually increased the size of the input segment from 21 frames (when the lower part of the network returns one frame-level output, so effectively no temporal pooling occurs) to 164 frames (2 seconds).

Table 1 shows the speaker recognition error rates on the development set of the 50-speaker corpus for different durations of temporal pooling. As we expected, we obtained a very low error rate already with just a single-frame output, which shows that the UTI videos are very speaker-specific. Gradually increasing the segment length and thus the number of aggregated frames improves the results, but the improvements are smaller than those obtained, for example, in language recognition [39].

5.2. Validating the x-vectors on an independent speaker set

The second experiment sought to test the generalization ability of the x-vector. We were worried about overfitting the training speakers, as the popular x-vector implementations are trained with orders of magnitudes more training data and typically with more than a thousand speakers [27]. Thus, to validate our x-vector DNN, we performed a speaker recognition experiment on the dataset of the 31 speakers not used during x-vector training.

Using the speaker embeddings to recognize or verify a new set of speakers is not trivial, as the speakers will be different from those seen during training. For optimal performance, the embedding vectors are typically post-processed by factor analy-
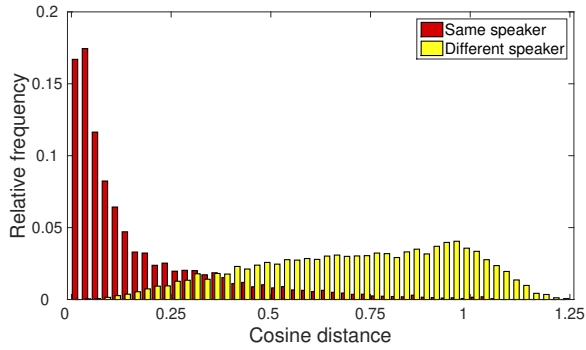
Figure 3: *Normalized histogram of the cosine distances for randomly chosen same-speaker and different-speaker x-vector pairs from the 31-speaker set.*

Table 2: *Speaker recognition error rates for the held-out 31 speakers using 1-NN leave-one-out testing.*

Embedding layer	*FC#1*		*FC#2*	
Activation	swish	linear	swish	linear
Error rate	0.96%	0.96%	2.03%	0.70%

Table 3: *Mean squared error (MSE) for the SSI spectral estimation task in the single-speaker and multi-speaker scenarios.*

SSI Train+Test configuration	Size of training set (frames)	MSE	
		Dev	Test
single-speaker	254306	0.256	0.265
multi-speaker	305040	0.603	0.669
multi-spk + Xvec		0.589	0.653

sis or dimension reduction methods [27, 40]. A simple and fast, though somewhat sub-optimal solution is to calculate the similarity score of two speakers based on the cosine similarity of their embedding vectors [40]. According to this, we performed a simple speaker recognition experiment with a 1-nearest neighbor classifier using the cosine distance as follows. We fused the development and the test sets of the 31 speakers, and calculated the x-vectors for each segment. Then we performed leave-one-out classification, that is, to each vector in this set we found the closest one (excluding itself). The classification was considered correct if the speaker IDs of the two segments were identical. We intentionally used the simplest possible 1-NN classifier, as it is very sensitive to the accuracy of the distance function applied, and hence to the accuracy of the underlying x-vectors.

Snyder et al. defined the x-vector as the linear output of the first fully connected layer after statistical pooling [27]. However, some authors questioned whether this is the optimal strategy, and also whether dropping the nonlinearity is required [34]. So we also examined which fully connected layer gives the best result, and if the nonlinearity is necessary or not. The results are shown in Table 2. As can be seen, all embedding extraction methods resulted in very low speaker recognition error rates around 1-2%. The best result was obtained when the speaker embedding was produced by the lower fully connected layer (FC#2) without the nonlinearity. This coincides with the findings of Snyder et al. [27] and Shon et al. [34].

To further demonstrate the speaker discriminative abilities of the x-vectors on a new set of speakers, a histogram of the cosine distances is shown in Fig 3 for 10000 randomly selected same-speaker and different-speaker vector pairs from the 31-speaker dev+test subset. Although the distributions overlap slightly, this figure also suggests that the embedding vectors behave as expected, that is, vectors from the same speaker are closer to each other than vectors from different speakers.

5.3. Applying the speaker embedding in speech synthesis

In the last experiment we attempted to use the x-vector as an auxiliary input for the spectral estimator network of our SSI system (cf. Fig 1). As the baseline, we trained the net with the single-speaker TaL1 corpus. The obtained mean squared error rates (MSE) shown in Fig 3 are around 0.26, which results in a speech signal with a mel-cepstral distortion (MCD) of 3.12 after speech synthesis. This corresponds to a low-quality, but intelligible speech [5]. For multi-speaker training and testing we used samples from the 31-speaker subset. We did not use all

the available data from the 31 speakers to demonstrate how the accuracy of the SSI network drops when switching to a multi-speaker scenario with a similar amount of training data. Table 3 shows that the multi-speaker setup led to a drastically larger MSE. We note that the multi-speaker scenario is still speaker-dependent in the sense that the training, development and test sets are from the same speakers. Even larger performance drops can be expected in a speaker-independent configuration [24].

Finally, we re-trained the multi-speaker model with the x-vector as auxiliary input. However, the optimal way of combining the ultrasound input with the x-vector is not trivial. As the ultrasound images are processed by convolution, simple concatenation was not an option. We decided to inject the x-vector into the network only after the convolutional layers, which were initialized by transfer learning from the multi-speaker model. However, this solution might be suboptimal and requires further studies. We mention that Ribeiro et al. used the mean ultrasound frame to represent the speaker, so they could add this image to the CNN input as a second channel [24].

The bottom row of Table 3 shows that although the introduction of the x-vector resulted in a consistent improvement for both the development and the test sets, this improvement is marginal. Ribeiro et al. reported similarly small gains from using the speaker mean as the speaker-characteristic input [24].

6. Conclusions

Here, we adjusted the x-vector framework of speech processing to ultrasound tongue videos to create a speaker-characteristic embedding vector. We modified the network architecture to process videos as input, and trained the network for speaker recognition. We obtained very low speaker recognition error rates, and our embedding vectors also seem to generalize well to new speakers. However, our first attempts to apply the embedding vector in a multi-speaker SSI scenario resulted in just a minimal improvement, showing that further studies are required on the proper application of the x-vector in this field.

7. Acknowledgements

8. References

[1] B. Denby, T. Schultz, K. Honda, T. Hueber, J. M. Gilbert, and J. S. Brumberg, "Silent speech interfaces," *Speech Communication*, vol. 52, no. 4, pp. 270–287, 2010.

[2] T. Schultz, M. Wand, T. Hueber, D. J. Krusienski, C. Herff, and J. S. Brumberg, "Biosignal-based spoken communication: A survey," *IEEE/ACM Trans. ASLP*, vol. 25, no. 12, pp. 2257–2271, 2017.

[3] J. A. Gonzalez-Lopez, A. Gomez-Alanis, J. M. Martin Donas, J. L. Perez-Cordoba, and A. M. Gomez, "Silent Speech Interfaces for Speech Restoration: A Review," *IEEE Access*, vol. 8, pp. 177 995–178 021, 2020.

[4] T. G. Csapó, T. Grósz, G. Gosztolya, L. Tóth, and A. Markó, "DNN-based ultrasound-to-speech conversion for a silent speech interface," in *Proc. Interspeech*, 2017, pp. 3672–3676.

[5] T. G. Csapó, C. Zainkó, L. Tóth, G. Gosztolya, and A. Markó, "Ultrasound-based Articulatory-to-Acoustic Mapping with Wave-Glow Speech Synthesis," in *Proc. Interspeech*, 2020, pp. 2727–2731.

[6] M. S. Ribeiro, A. Eshky, K. Richmond, and S. Renals, "Silent versus modal multi-speaker speech recognition from ultrasound and video," in *submitted to Interspeech*, 2021.

[7] A. H. Shandiz, L. Tóth, G. Gosztolya, A. Markó, and T. G. Csapó, "Improving neural silent speech interface models by adversarial training," in *The International Conference on Artificial Intelligence and Computer Vision*. Springer, 2021, pp. 430–440.

[8] A. H. Shandiz and L. Tóth, "Voice activity detection for ultrasound-based silent speech interfaces using convolutional neural networks," *arXiv preprint arXiv:2105.13718*, 2021.

[9] Y. Yu, A. H. Shandiz, and L. Tóth, "Reconstructing speech from real-time articulatory mri using neural vocoders," *arXiv preprint arXiv:2104.11598*, 2021.

[10] J. Wang, A. Samal, J. Green, and F. Rudzicz, "Sentence recognition from articulatory movements for silent speech interfaces," in *Proc. ICASSP*, 2012, pp. 4985–4988.

[11] M. Kim, B. Cao, T. Mau, and J. Wang, "Speaker-Independent Silent Speech Recognition From Flesh-Point Articulatory Movements Using an LSTM Neural Network," *IEEE/ACM Trans. ASLP*, vol. 25, no. 12, pp. 2323–2336, 2017.

[12] F. Taguchi and T. Kaburagi, "Articulatory-to-speech conversion using bi-directional long short-term memory," in *Proc. Interspeech*, 2018, pp. 2499–2503.

[13] M. J. Fagan, S. R. Ell, J. M. Gilbert, E. Sarrazin, and P. M. Chapman, "Development of a (silent) speech recognition system for patients following laryngectomy," *Medical Engineering and Physics*, vol. 30, no. 4, pp. 419–425, 2008.

[14] J. A. Gonzalez, L. A. Cheah, A. M. Gomez, P. D. Green, J. M. Gilbert, S. R. Ell, R. K. Moore, and E. Holdsworth, "Direct speech reconstruction from articulatory sensor data by machine learning," *IEEE/ACM Trans. ASLP*, vol. 25, no. 12, pp. 2362–2374, 2017.

[15] J. A. Gonzalez-Lopez, M. Gonzalez-Atienza, A. Gomez-Alanis, J. L. Perez-Cordoba, and P. D. Green, "Multi-view Temporal Alignment for Non-parallel Articulatory-to-Acoustic Speech Synthesis," in *Proc. IberSPEECH*, 2021, pp. 230–234.

[16] L. Maier-Hein, F. Metze, T. Schultz, and A. Waibel, "Session independent non-audible speech recognition using surface electromyography," in *Proc. ASRU*, 2005, pp. 331–336.

[17] M. Janke, M. Wand, K. Nakamura, and T. Schultz, "Further investigations on EMG-to-speech conversion," in *Proc. ICASSP*, 2012, pp. 365–368.

[18] M. Wand, T. Schultz, and J. Schmidhuber, "Domain-Adversarial Training for Session Independent EMG-based Speech Recognition," in *Proc. Interspeech*, 2018, pp. 3167–3171.

[19] S. Stone and P. Birkholz, "Cross-speaker silent-speech command word recognition using electro-optical stomatography," in *Proc. ICASSP*, online, 2020, pp. 7849–7853.

[20] A. Ephrat and S. Peleg, "Vid2speech: Speech reconstruction from silent video," in *Proc. ICASSP*, 2017, pp. 5095–5099.

[21] D. Michelsanti, O. Slizovskaia, G. Haro, E. Gómez, Z.-H. Tan, and J. Jensen, "Vocoder-Based Speech Synthesis from Silent Videos," in *Proc. Interspeech*, 2020, pp. 3530–3534.

[22] J. Freitas, A. J. Ferreira, M. A. T. Figueiredo, A. J. S. Teixeira, and M. S. Dias, "Enhancing multimodal silent speech interfaces with feature selection," in *Proc. Interspeech*, 2014, pp. 1169–1173.

[23] G. Gosztolya, T. Grósz, L. Tóth, A. Markó, and T. G. Csapó, "Applying DNN Adaptation to Reduce the Session Dependency of Ultrasound Tongue Imaging-Based Silent Speech Interfaces," *Acta Polytechnica Hungarica*, vol. 17, no. 7, pp. 109–124, 2020.

[24] M. S. Ribeiro, A. Eshky, K. Richmond, and S. Renals, "Speaker-independent Classification of Phonetic Segments from Raw Ultrasound in Child Speech," in *Proc. ICASSP*, 2019, pp. 1328–1332.

[25] M. Ribeiro, J. Sanger, J.-X. Zhang, A. Eshky, A. Wrench, K. Richmond, and S. Renals, "Tal: a synchronised multi-speaker corpus of ultrasound tongue imaging, audio, and lip videos," *arXiv preprint arXiv:2011.09804*, 2020.

[26] A. Senior and I. Lopez-Moreno, "Improving DNN speaker-independence with I-vector inputs," in *Proc. ICASSP*, 2014, pp. 225–229.

[27] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust DNN embeddings for speaker recognition," in *Proc. ICASSP*, 2018, pp. 5329–5333.

[28] E. Tatulli and T. Hueber, "Feature extraction using multimodal convolutional neural networks for visual speech recognition," in *Proceedings of ICASSP*, 2017, pp. 2971–2975.

[29] R. Prenger, R. Valle, and B. Catanzaro, "Waveglow: A flow-based generative network for speech synthesis," in *Proc. ICASSP*, 2019, pp. 3617–3621.

[30] L. Tóth and A. H. Shandiz, "3D convolutional neural networks for ultrasound-based silent speech interfaces," in *Proc. ICAISC*. Springer, 2020, pp. 159–169.

[31] V. Peddinti, D. Povey, and S. Khudanpur, "A time delay neural network architecture for efficient modeling of long temporal contexts," in *Proc. Interspeech*, 2015, pp. 3214–3218.

[32] C. Li, X. Ma, B. Jiang, X. Li, X. Zhang, X. Liu, Y. Cao, A. Kannan, and Z. Zhu, "Deep Speaker: an End-to-End Neural Speaker Embedding System," *arXiv e-prints*, p. arXiv:1705.02304, 2017.

[33] D. Snyder, P. Ghahremani, D. Povey, D. Garcia-Romero, Y. Carmiel, and S. Khudanpur, "Deep neural network-based speaker embeddings for end-to-end speaker verification," in *Proc. (SLT)*, 2016, pp. 165–170.

[34] S. Shon, H. Tang, and J. R. Glass, "Frame-level speaker embeddings for text-independent speaker recognition and analysis of end-to-end model," in *Proc. SLT*, 2018, pp. 1007–1013.

[35] K. Okabe, T. Koshinaka, and K. Shinoda, "Attentive statistics pooling for deep speaker embedding," in *Proc. Interspeech*, 2018, pp. 2252–2256.

[36] Z. Y., T. Ko, D. Snyder, B. Mak, and D. Povey, "Self-attentive speaker embeddings for text-independent speaker verification," in *Proc. Interspeech*, 2018, pp. 3573–3577.

[37] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri, "A closer look at spatiotemporal convolutions for action recognition," in *Proc. CVPR*, 2018.

[38] L. Tóth, "Combining time- and frequency-domain convolution in convolutional neural network-based phone recognition," in *Proc. ICASSP*, 2014, pp. 190–194.

[39] D. Snyder, D. Garcia-Romero, A. McCree, G. Sell, D. Povey, and S. Khudanpur, "Spoken language recognition using x-vectors," in *Proc. Odyssey 2018 The Speaker and Language Recognition Workshop*, 2018, pp. 105–111.

[40] N. Dehak, R. Dehak, P. Kenny, N. Brummer, P. Ouellet, and P. Dumouchel, "Support vector machines versus fast scoring in the low-dimensional total variability space for speaker verification," in *Proc. Interspeech*, 2009, pp. 1559–1562.