BUT CHiME-7 system description

Martin Karafiát[1], *Karel Veselý*[1], *Igor Szőke*[1], *Ladislav Mošner*[1], *Karel Beneš*[1], *Marcin Witkowski*[2], *Germán Barchi*[3], *Leonardo Pepino*[3]

[1]Brno University of Technology
[2]AGH University of Krakow
[3]University of Buenos Aires

karafiat@fit.vutbr.cz, iveselyk@fit.vutbr.cz, szoke@fit.vutbr.cz, imosner@fit.vutbr.cz, ibenes@fit.vutbr.cz, witkow@agh.edu.pl, gbarchi@dc.uba.ar, lpepino@dc.uba.ar

Abstract

This paper describes the joint effort of Brno University of Technology (BUT), AGH University of Krakow and University of Buenos Aires on the development of Automatic Speech Recognition systems for the CHiME-7 Challenge. We train and evaluate various end-to-end models with several toolkits. We heavily relied on Guided Source Separation (GSS) to convert multi-channel audio to single channel. The ASR is leveraging speech representations from models pre-trained by self-supervised learning, and we do a fusion of several ASR systems. In addition, we modified external data from the LibriSpeech corpus to become a close domain and added it to the training.

Our efforts were focused on the far-field acoustic robustness sub-track of Task 1 - Distant Automatic Speech Recognition (DASR), our systems use oracle segmentation.

Index Terms: speech recognition, human-computer interaction

1. Introduction

This paper describes the BUT Automatic Speech Recognition (ASR) system for the CHiME-7 Speech to Text Transcription (STT) Challenge. We present the detailed description of the datasets, as well as technical details for the development of ASR subsystems and the fusion.

2. Data

Our training setup is derived from the released baseline ESPnet [1] recipe[1]. The training setup is composed from *Chime6* [2], *Mixer6* (LDC2013S03), and *Dipco* [3] datasets, where *Chime6* and *Mixer6* have training/dev/eval data split, and *Dipco* is used only for development and evaluation.

Table 1 describes all the training data selection setups used in our system building. The *baseline* and *baseline + mixer6gss* shows amount of training data used in baseline recipe before and after fixing bug in baseline recipe (missing GSS for *Mixer6* data). The target data are always processed with GSS, and the amount of GSS-processed training data is very limited in the baseline recipe. Therefore, we decided to modify the data preparation setup, and we created new data sets *limited* and *limited+libri* with following procedure:

- We added GSS-enhanced Mixer6 training data (see section 3.1), which were missing due to a bug in the baseline recipe.
- 3-way speed perturbation is applied only to the GSS data.

[1]https://github.com/espnet/espnet/tree/master/egs2/chime7_task1/asr1

Table 1: *Training data selections*

Dataset	# hours	GSS part
baseline	5922	1.7%
baseline + mixer6gss	6301	2.8%
limited	611	29.1%
limited+libri	1108	16.1%
gss-only	288	100%

- The data were re-balanced to increase the weight of the GSS-enhanced data. We randomly chose 80 hours of each combination of `(Chime6, Mixer6) x (IHM, MDM)` type of the training utterances.
- LibriSpeech [4] data augmented with *background speaker* were added.

The *limited* dataset in Table 1 consists of the speed-perturbed 'full amount' of the GSS-processed data and the subsampled IHM+MDM data. The *limited+libri* has the augmented LibriSpeech added, and the *gss-only* consists purely of speed-perturbed GSS-processed data (*Chime6*, *Mixer6*).

2.1. Multispeaker augmentation

We used LibriSpeech dataset to simulate 'multispeaker' condition. All recordings were augmented by another randomly selected recording, coming from the same subset of LibriSpeech, to insert a so called 'background speaker'. The background speaker was expanded by 4 seconds of silence on both sides. Then, this expanded audio was merged with the original audio file starting in random position (and the expanded audio was looped if the end of the background speaker was achieved sooner than the end of the original audio). We maintain the Signal-to-Noise Ratio (SNR) between these two audios in the range of 5dB - 12dB randomly. Each of the audio was reverberated by a single Room Impulse Response (RIR) randomly selected from a pool shown in Table 2. Finally, one of the following codecs (MP3, AMR, AMR-WB, G.711, G.726, G.729, GSM-FR, TETRA, GSM-EFR) was applied with a probability

Table 2: *Used RIRs*

Dataset	#RIRs
AIR14 [5]	214
REVERB [6]	192
RWCP [7]	3240
ReverbDB [8]	1364
Synthetic	10000

of 1/7 on the resulting merged audio. The resulted merged audio sounds like the original LibriSpeech in a reverberated environment with random speaker speaking in the background. Synthetic RIRs were generated using the image method [9]. We randomly sampled the dimensions of the room as $width = [1.5 - 5.5]$, $height = [2.0 - 9.5]$, $length = [2.5 - 16.5]$. The source and microphone were randomly placed in the room and the wall reflections were anything between $\beta = [0.45 - 0.95]$. We used only a 'visible' subset of RIRs from ReverbDB [8].

3. Speech Enhancement

3.1. Non-neural approach - GSS

The speech enhancement was based on the baseline system provided by the organizers. First, $k = 80\%$ channels were selected for further processing using the Envelope Variance [10] method. The baseline Guided Source Separation (GSS) [11] was used as a primary speech enhancement technique, which integrates Weighted Prediction Error (WPE) dereverberation method [12], estimation of target and undesired component time-frequency masks using oracle diarization output and Complex Angular Central Gaussian Mixture Model [13] posteriors, and the mask-based Minimum-Variance Distortionless Response (MVDR) beamformer [14]. The GPU-based implementation of GSS [11] with its default parameters was used in the experiments as a baseline. Table 3 contains the Word Error Rates (WER) results obtained using different non-neural speech enhancement methods for the pretrained ASR model on the development sets used in the Challenge. Note, the pretrained models was WAVLM based model trained with baseline recipe and uploaded by organizers. The running script was simply called with `--use-pretrained popcornell/chime7_task1_asr1_baseline` option to quickly analyze speech enhancement technique.

To validate the performance of stronger filtering, we experimented with reuse of CACGMM estimated target masks as a post-filter applied to the output of the beamformer. This result is presented in the second row of Table 3. To investigate the performance of using the filter that allows for milder attenuation than using masks directly, we have also experimented with the use of a Convolutional Weighted Multichannel Wiener filter (CWMWF) [15] as a convolutional beamformer which performs dereverberation and source separation jointly. The general diagram of the investigated systems is depicted in Figure 1.

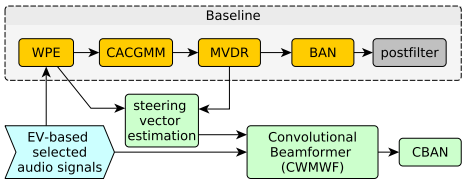

Figure 1: *Evaluated non-neural speech enhancement.*

A steering vector was selected as the principal eigenvector of the spatial covariance matrix (SCM) calculated using the multiplication of the multichannel output of WPE and the estimated MVDR beamformer from the baseline system. This steering vector and multichannel input were used to estimate CWMWF. A Convolutional Blind Analytical Normalization (CBAN) was applied to the output signal to compensate for attenuation introduced by the beamformer. The weighting factors for a single frequency bin k were calculated similarly to non-convolutional Blind Analytical Normalization (BAN) [16] with CWMWF coefficients $\mathbf{w}(k) \in \mathbb{C}^{IL}$ and convolutional noise covariance matrix $\mathbf{\Phi}_N(k) \in \mathbb{C}^{IL \times IL}$, where I, L are the number of channels and the number of filter taps, respectively. For audio segments in which the maximum signal amplitude after analytical normalization exceeded one, additional peak normalization was applied to prevent clipping.

Table 3: *WER [%] using the pretrained baseline ASR with different non-neural speech enhancement on development sets.*

Method	Chime6	Dipco	Mixer6
GSS (the baseline)	33.3	34.6	21.8
GSS (baseline+postfilter)	35.8	36.8	23.5
GSS (context 1s)	47.4	56.8	24.6
GSS+CWMWF+CBAN	49.5	58.4	26.9

In the baseline system, the multichannel signal is extended by the left and right context of 15 seconds by default to compute WPE dereverberation filter and estimate masks with CACGMM. Then the context is dropped when calculating MVDR beamformer coefficients. Since the convolutional filter performs joint dereverberation and separation the context for this filter has been greatly reduced to minimize artifacts resulting from estimation filter from longer segments. So far in the experiments we have shown that after reduction of the context to 1 second the CWMWF achieves comparable results to the baseline system. Note that in those experiments, ASR system was not fine-tuned for the new domain of audio data introduced by stronger filtering. Therefore, we claim that presentation of the processed data to the ASR system is crucial.

3.2. Neural approach

Contrary to GSS-based pre-processing, we experimented with discriminative models. A general structure of the approach is depicted in Figure 2. In the first stage, we employ Target Speaker Extraction (TSE) models to provide per-channel estimates of the speaker-of-interest speech where the enrollment utterances are provided by the oracle diarization. Given the predictions, input audio, and the assumption that speech and noise are not correlated, we estimate noise by subtracting speech from mixtures. Subsequently, both speech and noise signals are transformed via STFT to provide ratio masks. A speech mask is computed as a ratio between the power spectrum of predicted speech and the sum of the power spectra of speech and noise. A noise mask is obtained analogically. We follow a standard approach to estimate spatial covariance matrices of speech and noise using the masks [17]. They are used to compute beamforming weights following Minimum Variance Distortionless Response (MVDR) approach [18, 19], which combine input channels previously selected analogically to the baseline GSS.

We experimented with three types of models for TSE — SpeakerBeam [20], DPCCN [21], and a down-scaled version of TF-GridNet [22] conditioned on the enrollment utterance through Feature-wise Linear Modulation (FiLM) [23]. The TF-GridNet-based TSE was inspired by that used in iNeuBe-X [24]. Therefore, we also use the encoder and TCN modules of TCN-DenseNet [25] to extract speaker embedding from the enrollment segments. Compared to TSE in iNeuBe-X, multiple channels are not concatenated to form the input, but for consistency with other explored models, each channel is processed independently. For computation reasons, we set number of blocks B=5 and LSTM hidden units H=128 in TF-GridNet. The number of

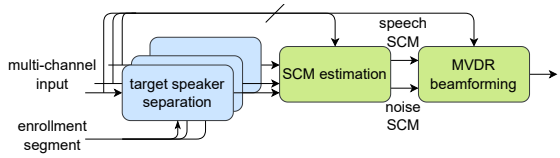

Figure 2: *Multi-channel speech enhancement based on neural networks.*

Table 4: *WER [%] using the pretrained baseline ASR with different neural speech enhancement on development sets.*

Method	Mixing SNR	Chime6	Dipco	Mixer6
SpeakerBeam	5 – 30	54.4	50.4	21.6
DPCCN	5 – 30	50.9	48.6	21.6
TF-GridNet	5 – 30	50.9	49.8	21.8
SpeakerBeam	−4 – −10	52.6	47.4	21.3

repeats in TCNDenseNet was reduced to 2. For SpeakerBeam and DPCCN, we used hyperparameters following the respective papers.

All the TSE networks were trained with the SNR objective. Since the reference speech signals are required during training, we dynamically simulated the data. As a source of speech, we used the train-clean-360 subset of Librispeech. As a source of background noise, MUSAN and WHAM [26] datasets were utilized. The SNR intervals for adding noise are shown in Table 4. To simulate room acoustic conditions, we reverberated the signals with simulated room impulse responses (with RT60 from 0.3 to 0.9) and those drawn from the ReverbDB corpus [8]. In addition to the target speaker, audio mixtures contain up to 2 other voices, while only the target speaker's speech examples are also allowed.

The comparison of the TSE methods on the development sets is presented in Table 4, this is with the baseline ASR system employed. We observed noticeable degradation for Chime6 and Dipco compared to the baseline GSS. We hypothesize it is mainly caused by the domain mismatch of test and training data and the inability of neural models to generalize. While poor generalization of time-domain models is common, we did not observe improvements from using TF-GridNet. However, we can confirm that TF-GridNet provides better predictions compared to other models as we observed differences in results when experimenting with the approach where SCMs were computed directly using network outputs (without masking).

4. Automatic Speech Recognition systems

We experimented with Encoder-Decoder ASR models coming from various toolkits. Some ASR systems are built on top of 'fixed' pre-trained feature representations, while in other cases these pre-trained models are 'fine-tuned'.

4.1. HuggingFace

We fine-tuned the WavLM large model by adding a conformer layer and a RNN transducer [27] on top of it. The conformer layer consists of 4 attention heads, a kernel size of 31, downsampling in time by a factor of 4 and dropout with rate 0.1. These are the default values of the torchaudio implementation we used. The predictor network consists of 2 LSTM layers and we also used the RNNT loss from torchaudio implementation. The model was fine-tuned using GSS data from Chime6 and Mixer6 datasets (i.e. *gss-only* in Table 1). The CNN encoder of the WavLM model was frozen during the fine-tuning, updating only the weights from the transformer layers and the RNNT modules. We applied a learning rate of 1e-4 with a warmup of 1000 steps, and a batch size of 16 samples. The model was trained for 30000 steps using the AdamW optimizer with a weight decay of 0.005. For the generation of the transcriptions, we used the RNNT beam search decoder from torchaudio with a beam size of 20.

4.2. WavLM with speaker information

We also explored adding the target speaker information to the WavLM model. This could potentially help the model to focus on transcribing the target speaker when the GSS signal contains interference from non-target speakers. We used x-vectors from VBx as speaker embeddings, linearly projected them and summed them to the input of the first transformer layer. We learned a scale factor for this projected embedding, which was initialized to 0 so that at the beginning of training, no speaker information is added and the model is equivalent to the original WavLM. We verified that during training, the scale factor increased in magnitude, suggesting that the model is in fact using the target speaker information. In order to train the model with a larger amount of speakers, we augmented the training data with artificial mixtures of up to 3 speakers from Librispeech, resulting in around 200k extra utterances.

4.3. EspNET

We selected two pre-trained models via S3PRL `wav2vec2_large_lv60_cv_swbd_fsh` [28] and `wavlm_large` [29] for building systems with ESPnet toolkit.

WavLM Large + Transfomer This system followed the baseline recipe, therefore the initial *wavlm_large* model is followed by Transformer based model: encoder (12 layers) – decoder (6 layers).

Wav2Vec2 + Conformer This system is based on the Conformer architecture [30] and is composed of 12 encoder layers and 6 decoder layers. The conformer encoder layer incorporates, in addition to a self-attention module, a convolutional layer in between of two feed-forward modules. The decoder was built using masked self-attention as well as cross-attention between the encoder embeddings and the decoder. Each encoder and decoder layer outputs 512 dimensional embeddings; attention is done with 8 parallel heads and the feed-forward module expands the data into 2048 dimensions. We used the standard ESPnet2 training recipe with 40k warm-up steps and learning rate $2 \cdot 10^{-3}$.

Both models were trained with frozen update of the pre-trained models (WavLM, Wav2Vec2). The model output are 500 Sentencepiece [31] unigram units. The model is trained with the joint CTC/Attention loss with the CTC weight of 0.3.

The systems were further fine-tuned with the *gss-only* data (in Table 1) using lower learning rate, for this step we 'defreezed' weights in the pre-trained models (WavLM, Wav2Vec2) from S3PRL.

4.4. K2

The K2 codebase was extended to accomodate S3PRL models, such as *wavlm-large* [29], as a fixed feature extraction. The *wavlm-large* model transforms raw audio signal to 1024 dimensional embeddings with 20ms time-shift. We duplicate each em-

Table 5: *Word Error Rates [%] obtained on dev parts of the datasets for various system architectures and training data.*

	Pre-trained model	fine-tuned	Architecture (toolkit)	Training data	Enh.	Chime6	Dipco	Mixer6
0	WavLM Large	✗	Transformer-Transformer (ESPnet)	*baseline*	GSS	33.5	35.4	23.7
1	WavLM Large	✗	Transformer-Transformer (ESPnet)	*limited*	GSS	32.2	33.2	21.1
$2^{\dagger}$	WavLM Large	✗	Transformer-Transformer (ESPnet)	*limited+libri*	GSS	30.2	31.9	19.9
2(S)	WavLM Large	✓	Transformer-Transformer (ESPnet)	($2^{\dagger}$) →*gss-only*	GSS	**25.4**	**29.8**	18.7
$3^{\dagger}$	WavLM Large	✗	Transformer-Transformer (ESPnet)	*limited+libri*	GSS_postfilter	32.9	33.5	21.8
3	WavLM Large	✓	Transformer-Transformer (ESPnet)	($3^{\dagger}$) →gss-only	GSS_postfilter	26.6	31.1	19.6
$4^{\dagger}$	WavLM Large	✓	Conformer-Transformer (ESPnet)	*limited+libri*	GSS	29.2	30.9	17.4
4	WavLM Large	✓	Conformer-Transformer (ESPnet)	($4^{\dagger}$) →gss-only	GSS	25.7	30.1	16.9
$5^{\dagger}$	Wav2Vec2 Large	✗	Conformer-Transformer (ESPnet)	*limited+libri*	GSS	37.9	40.3	21.7
5	Wav2Vec2 Large	✓	Conformer-Transformer (ESPnet)	($5^{\dagger}$) →*gss-only*	GSS	31.3	38.2	20.2
6	WavLM Large	✓	CTC (HuggingFace)	*gss-only*	GSS	40.2	50.4	28.7
7	WavLM Large	✓	Conformer-Transducer (HuggingFace)	*gss-only*	GSS	28.0	36.0	17.2
8	WavLM Large	✓	Conformer-Transducer (HuggingFace)	*gss-only*	GSS	26.3	31.7	**15.8**
9	WavLM Large	✓	LinSum XVector	*librimix+gss*	GSS	28.4	34.1	17.8
10	WavLM Large	✗	Zipformer-Transducer (K2)	*limited*	GSS	30.0	32.5	17.5
F1	–		2 + 8	–	GSS	25.3	31.2	18.3
F2	–		4 + 8	–	GSS	25.0	30.2	16.0
F3	–		3 + 4 + 8	–	–	**23.8**	**28.8**	**15.4**

bedding vector to create a stream with 10 ms time-steps.

On top of these embeddings, we trained a *Zipformer-Transducer* model from the streaming ASR recipe [2]. Due to using a pre-encoder, we removed the `Conv2dSubsampling` front-end module in Zipformer and we reduced the numbers of `ZipformerEncoderLayer` modules to : `[1,1,1,1,1]`. To link the *wavlm-large* pre-encoder to *Zipformer encoder*, a trainable linear transform without bias was introduced to reduce the dimensionality from 1024 to 384 dimensions. The model is using a *stateless* Predictor network in Transducer architecture [32]. And, the training is accelerated by a *pruning* secondary output-layer that pre-selects the candidate tokens [33]. The Zipformer-Transducer model uses half-precision, the model size is 386 MB, and the output are 500 sentence-piece [31] unigram units. The *wavlm* pre-econder had frozen parameters.

We trained for 15 epochs with base learning-rate 0.025, and we observed an increased level of overfitting. The overfitting was apparent for the `valid_pruned_loss` objective, starting after the 4th epoch. We used the *limited* dataset from Table 1. We did not use SpecAugment because of using the S3PRL feature transform pre-encoder.

For decoding we used the `fast_beam_search_nbest` method with default hyper-parameters. For the final fusion, we exported N-best lists of size up to 200 generated by sampling a lattice.

4.5. System fusion

To facilitate effective fusion of outputs of the different systems, we first compact each resulting N-best list into a CTM using the Hystoc tool [34] with the temperature parameter set to 1.0. To merge the CTMs, we used NIST Rover [35], where the selection of output words is done according to the word frequency and maximum confidence. In Rover, we tuned the α parameter, which is a trade-off between frequency of word occurrence and maximum word confidence, as well as the null word confidence (also known as blank symbol confidence). In specific, α and null word confidence were set to 0.8 and 0.4, respectively (for the best performing fusion F3). We did not use the time information during fusion.

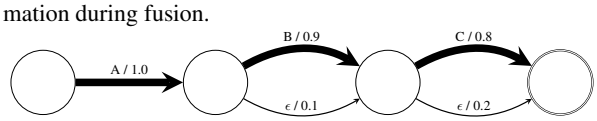

Figure 3: *Hystoc confidences for n-best set ABC, AB, and AC with probabilities 0.7, 0.2, and 0.1 respectively.*

5. Results

The results are presented in Table 5, in which we indicate the pre-trained feature transform (WavLM Large or Wav2Vec2Large), whether the feature transform was fixed or fine-tuned, the encoder-decoder parts of the ASR model on top of that transform (e.g. Trasformer-Transformer), and the ASR toolkit that was used. The training data indicators refer to Table 1 and for example '($3^{\dagger}$) →*gss-only*' indicates that the system was trained as system ($3^{\dagger}$) and then fine-tuned using the *gss-only* data. Development sets were enhanced with the baseline GSS for almost all systems, except for systems (3) and ($3^{\dagger}$) where 'GSS_postfilter', i.e. GSS with enabled mask-based post-filtering, was used.

Our *limited* training data selection presented above (system (1)) in Section 2 is giving 1.3–2.6% absolute gain over baseline (0) with significant increase of training speed as the *limited* set contains 10x less data than *baseline* one. Moreover, next 1.2–2.0% absolute gain is reached with adding 'enhanced' Librispeech data for system ($2^{\dagger}$) that was trained with *limited+libri* dataset. Additional fine-tuning of this system to the *gss-only* data is giving further significant 1.2–4.8% absolute WER reduction resulting in system (2(S)). The system (2(S)) was **submitted** as the single best system to the challenge.

The comparison of systems ($2^{\dagger}$) and ($3^{\dagger}$) shows a drop in

[2] `https://github.com/vesis84/icefall/tree/master/egs/librispeech/ASR/pruned_transducer_stateless7_streaming`

ASR performance similar to the one observed in Table 3, the difference is the reuse use of the CACGMM TF masks on the output of MVDR beamformer as *GSS_postfilter* in ($3^{\dagger}$). Again, the significant drop of WER from ($3^{\dagger}$) to (3) confirms the great importance of fine-tuning with the in-domain data.

The WavLM based systems were found superior to Wav2Vec2 systems, see comparison (4) and (5). Replacing the CTC loss by a RNN-T loss brought large improvements in WER, as seen in the gap between systems (6) and (7), showing the importance of incorporating a language model for this task. Moreover, when also fine-tuning the CNN encoder in WavLM, and lowering the learning rate from 1e-4 to 1e-5, we observe significant WER improvements between systems (7) and (8). Conditioning WavLM with speaker information didn't bring improvements as seen between systems (8) and (9). More work remains to be done to see if improved results can be achieved by using other conditioning methods like Adaptive Instance Normalization and FiLM layers.

Next, we trained an ASR system with the K2 toolkit using Zipformer-Transducer architecture (system (10)). Unfortunately, we did not have time to finalize this work for the system presentation at the workshop. However, this work could be very promising for future, as it significantly outperforms system (1) trained on same data, while both systems are without fine-tuning the WavLM model.

Fusing the best performing systems (F1–F3) did provide modest gains over the individual best systems. Despite the performance drop observed on the development set for system (3) with *GSS_postfilter*, in the fusion (F3) the system (3) brings complementary information, improving the result compared to fusion (F2). This phenomenon might be justified by the fact that for some part of the data-set the stronger filtering is actually helpful.

6. Acknowledgements

This work has been funded by the EU's Horizon 2020 research and innovation program under the Marie Sklodowska-Curie grant agreement No 101007666, the Agency is not responsible for this results or use that may be made of the information. This research project has been also supported by the program "Excellence initiative – research university" for AGH University of Krakow.

7. References

[1] S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishitoba, Y. Unno, N. Enrique Yalta Soplin, J. Heymann, M. Wiesner, N. Chen, A. Renduchintala, and T. Ochiai, "ESPnet: End-to-End Speech Processing Toolkit," in *Proceedings of Interspeech*, 2018, pp. 2207–2211. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2018-1456

[2] S. Watanabe, M. Mandel, J. Barker, E. Vincent, A. Arora, X. Chang, S. Khudanpur, V. Manohar, D. Povey, D. Raj, D. Snyder, A. Subramanian, J. Trmal, B. Yair, C. Boeddeker, Z. Ni, Y. Fujita, S. Horiguchi, N. Kanda, and N. Ryant, "Chime-6 challenge: Tackling multispeaker speech recognition for unsegmented recordings," 05 2020, pp. 1–7.

[3] M. V. Segbroeck, Z. Ahmed, K. Kutsenko, C. Huerta, T. Nguyen, B. Hoffmeister, J. Trmal, M. Omologo, and R. Maas, "Dipco - dinner party corpus," in *Interspeech 2020*, 2019. [Online]. Available: https://www.amazon.science/publications/dipco-dinner-party-corpus

[4] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An asr corpus based on public domain audio books," 04 2015, pp. 5206–5210.

[5] M. Jeub, M. Schafer, and P. Vary, "A Binaural Room Impulse Response Database for the Evaluation of Dereverberation Algorithms," in *2009 16th International Conference on Digital Signal Processing*, July 2009, pp. 1–5.

[6] K. Kinoshita, M. Delcroix, T. Yoshioka, T. Nakatani, A. Sehr, W. Kellermann, and R. Maas, "The REVERB Challenge: A Common Evaluation Framework for Dereverberation and Recognition of Reverberant Speech," in *2013 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, Oct 2013, pp. 1–4.

[7] S. Nakamura, K. Hiyane, F. Asano, T. Nishiura, and T. Yamada, "Acoustical Sound Database in Real Environments for Sound Scene Understanding and Hands-free Speech Recognition." in *LREC*, 2000.

[8] I. Szöke, M. Skácel, L. Mošner, J. Paliesek, and J. Černocký, "Building and evaluation of a real room impulse response dataset," *IEEE Journal of Selected Topics in Signal Processing*, vol. 13, no. 4, pp. 863–876, 2019.

[9] E. A. P. Habets, "Room Impulse Response Generator," Tech. Rep., September 2010. [Online]. Available: https://github.com/ehabets/RIR-Generator

[10] M. Wolf and C. Nadeu, "Channel selection measures for multi-microphone speech recognition," *Speech Communication*, vol. 57, pp. 170–180, 2014.

[11] D. Raj, D. Povey, and S. Khudanpur, "GPU-accelerated guided source separation for meeting transcription," *arXiv preprint arXiv:2212.05271*, 2022.

[12] T. Yoshioka and T. Nakatani, "Generalization of multi-channel linear prediction methods for blind mimo impulse response shortening," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 10, pp. 2707–2720, 2012.

[13] N. Ito, S. Araki, and T. Nakatani, "Complex angular central Gaussian mixture model for directional statistics in mask-based microphone array signal processing," in *2016 24th European Signal Processing Conference (EUSIPCO)*. IEEE, 2016, pp. 1153–1157.

[14] M. Souden, J. Benesty, and S. Affes, "On optimal frequency-domain multichannel linear filtering for noise reduction," *IEEE Transactions on audio, speech, and language processing*, vol. 18, no. 2, pp. 260–276, 2009.

[15] M. Fraś, M. Witkowski, and K. Kowalczyk, "Convolutive Weighted Multichannel Wiener Filter Front-end for Distant Automatic Speech Recognition in Reverberant Multispeaker Scenarios," in *Interspeech 2022*, 2022.

[16] E. Warsitz and R. Haeb-Umbach, "Blind acoustic beamforming based on generalized eigenvalue decomposition," *IEEE Transactions on audio, speech, and language processing*, vol. 15, no. 5, pp. 1529–1539, 2007.

[17] J. Heymann, L. Drude, and R. Haeb-Umbach, "Neural Network Based Spectral Mask Estimation for Acoustic Beamforming," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 196–200.

[18] J. Capon, "High-resolution Frequency-wavenumber Spectrum Analysis," *Proceedings of the IEEE*, vol. 57, no. 8, 1969.

[19] M. Souden, J. Benesty, and S. Affes, "On Optimal Frequency-Domain Multichannel Linear Filtering for Noise Reduction," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 2, 2010.

[20] K. Žmolíková, M. Delcroix, K. Kinoshita, T. Ochiai, T. Nakatani, L. Burget, and J. Černocký, "SpeakerBeam: Speaker Aware Neural Network for Target Speaker Extraction in Speech Mixtures," *IEEE Journal of Selected Topics in Signal Processing*, vol. 13, no. 4, pp. 800–814, 2019.

[21] J. Han, Y. Long, L. Burget, and J. Černocký, "DPCCN: Densely-Connected Pyramid Complex Convolutional Network for Robust Speech Separation and Extraction," in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 7292–7296.

[22] Z.-Q. Wang, S. Cornell, S. Choi, Y. Lee, B.-Y. Kim, and S. Watanabe, "TF-GRIDNET: Making Time-Frequency Domain Models Great Again for Monaural Speaker Separation," in *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023, pp. 1–5.

[23] E. Perez, F. Strub, H. de Vries, V. Dumoulin, and A. Courville, "FiLM: Visual Reasoning with a General Conditioning Layer," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, Apr. 2018.

[24] S. Cornell, Z.-Q. Wang, Y. Masuyama, S. Watanabe, M. Pariente, and N. Ono, "Multi-Channel Target Speaker Extraction with Refinement: The Wavlab Submission to the Second Clarity Enhancement Challenge," *arXiv preprint arXiv:2302.07928*, 2023.

[25] Z.-Q. Wang, G. Wichern, and J. L. Roux, "Leveraging Low-Distortion Target Estimates for Improved Speech Enhancement," *arXiv preprint arXiv:2110.00570*, 2021.

[26] G. Wichern, J. Antognini, M. Flynn, L. R. Zhu, E. McQuinn, D. Crow, E. Manilow, and J. Le Roux, "Wham!: Extending speech separation to noisy environments," in *Proc. Interspeech*, Sep. 2019.

[27] A. Graves, "Sequence transduction with recurrent neural networks," *CoRR*, vol. abs/1211.3711, 2012. [Online]. Available: http://arxiv.org/abs/1211.3711

[28] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations," in *proceedings of NeurIPS 2020*, 2020.

[29] S. Chen, C. Wang, Z. Chen, Y. Wu, and et al., "WavLM: Large-Scale Self-Supervised Pre-Training for Full Stack Speech Processing," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 6, pp. 1505–1518, July 2022.

[30] A. Gulati, J. Qin, C. C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu, and R. Pang, "Conformer: Convolution-augmented transformer for speech recognition," *arXiv preprint arXiv:2005.08100*, 2020.

[31] T. Kudo and J. Richardson, "Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018: System Demonstrations, Brussels, Belgium, October 31 - November 4, 2018*. Association for Computational Linguistics, 2018.

[32] M. Ghodsi, X. Liu, J. Apfel, R. Cabrera, and E. Weinstein, "Rnn-transducer with stateless prediction network," in *ICASSP 2020*. IEEE, 2020.

[33] F. Kuang, L. Guo, W. Kang, L. Lin, M. Luo, Z. Yao, and D. Povey, "Pruned RNN-T for fast, memory-efficient ASR training," in *Interspeech 2022*, H. Ko and J. H. L. Hansen, Eds. ISCA, 2022.

[34] K. Beneš, M. Kocour, and L. Burget, "Hystoc: Obtaining word confidences for fusion of end-to-end asr systems," 2023.

[35] J. Fiscus, "A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (rover)," in *1997 IEEE Workshop on Automatic Speech Recognition and Understanding Proceedings*, 1997, pp. 347–354.