Università Politecnica delle Marche
Scuola di Dottorato di Ricerca in Ingegneria dell' Informazione
Curriculum in Ingegneria Informatica, Gestionale e dell' Automazione

Front-End Processing for Speech Applications with Deep Learning Techniques

Ph.D. Dissertation of:
Samuele Cornell

Advisor:
Prof. Stefano Squartini

Curriculum Supervisor:
Prof. Franco Chiaraluce

XXXV ciclo - nuova serie

Università Politecnica delle Marche
Scuola di Dottorato di Ricerca in Ingegneria dell' Informazione
Facoltà di Ingegneria
Via Brecce Bianche – 60131 Ancona (AN), Italy

Acknowledgments

It almost feel like a lifetime has passed in these three years. Not to lie, this PhD journey has been intense and, as I am near its completion, I am grateful for having had the opportunity of learning so much and meeting so many wonderful people, many of which I now consider among my dearest friends. It is often said that's the journey that matters not the destination. But it is the people you meet and share the journey with what truly makes it worthwhile.

First and foremost I would like to thank my family and my parents for their unconditional love and support especially during my first two PhD years, which have been especially difficult due to the global pandemic.

My heartfelt gratitude to my advisor Stefano Squartini, who introduced me this wonderful research field. His constant motivation, enthusiasm, support and guidance have been indispensable to keep me going. Speech processing is extremely fascinating but is also very challenging. Its exponential growth, witnessed in recent years, makes very easy to get lost into this maelstrom of new technologies and applications. Mastering this subject is not an easy feat and demands a lot of hard work, dedication, and continuous learning and, in these three years, I feel I have only scraped the tip of the iceberg.

To my friends and collaborators, this dissertation is as much as yours as it is mine, since all the works here would have not been possible without all of you. I would like to thank Maurizio Omologo, Mirco Ravanelli and the John Hopkins University for giving me the opportunity of participating in the JSALT 2019 Summer workshop. A special thanks also to Yenda Trmal and Najim Dehak for having managed its organization. It has been a life-changing experience for me (really!), those two months convinced me to work in this field and have been by far the most intense learning experience I ever had in my lifetime. I had a great time and fruitful collaboration with so many people there: Manuel Pariente, Santiago Pascual, Alessio Brutti, Emmanuel Vincent, Tobias Menne, Tina Raissi and Sunit Sivasankaran. And indeed some of the works in this thesis came from ideas we developed together in those two wonderful months in Montreal. My friend Manuel Pariente especially, is the one that introduced me to the field of speech separation and our collaboration spanned all these three years, and includes, most notably, the development of Asteroid. I can't thank him enough.

Under the supervision of Emmanuel Vincent and Maurizio Omologo I started at JSALT my work on overlapped speech detection and counting. I owe to them most of my knowledge of signal processing. Under their guidance I learnt a lot also how to conduct and present research. In Montreal I had an amazing time with Alessio Brutti and our idea of using learning to rank techniques for channel selection was born there. He also introduced me to diarization and we worked together on this matter during the AGEVOLA project. Two of the total five works presented in this dissertation were done with him.

Beside the JSALT, I owe to Mirco Ravanelli also the opportunity of joining the amazing SpeechBrain team, where I was able to grow also in my programming skills and had a great time with Cem Subakan, Titouan Parcollet, Francois Grondin and Aku Rouhe.

I would like to thank also Romain Serizel, Nicolas Turpault and Francesca Ronchini for the opportunity of co-organizing the DCASE 2021/2022 Task 4 Challenge. I am looking forward to collaborate with you again in the field of machine listening.

In summer 2021 I was offered an internship at the Amazon Wakeword team in Cambridge where I found two incredible colleagues, Thomas Balestri and Thibaud Sénéchal. I learnt a lot and it was a pleasure to work with you.

Especially warm thanks to Shinji Watanabe. I was in his lab in summer 2022 and I owe to him and Zhong-Qiu Wang much of the research done in this last year. There I had the opportunity to meet, collaborate and learn from other amazing researchers and friends: Yoshiki Masuyama, Xuankai Chang and Juncheng Li. I would like also to mention the ESPNet-SE team, and, in particular Zhaoheng Ni and Yen-Ju Lu. Also a warm greeting and many thanks to my CHiME-7 DASR co-organizers Matthew Wiesner, Desh Raj, Paola Garcia and Marc Delcroix. Naoyuki Kamo, thank you for the invaluable help with ESPNet and the baseline code.

Thanks finally to my labmates at UNIVPM, in particular to Michela Cantarini, Emanuele Principi, Giovanni Morrone, Carlo Aironi and Leonardo Gabrielli. Due to the pandemic, I did not hang out too much in the lab but when I did I always had a great time with you all.

In closing, it is all thanks to each one of you, if, in these three years, I was able to grow both as a researcher and, most importantly, as a human being. Working and collaborating with all of you has been a privilege and I am humbled by the achievements that we have accomplished together.
Thank you for being a part of this journey.

Ancona, Maggio 2023

Samuele Cornell

Abstract

Front-end speech processing plays a vital role in many everyday applications such as teleconferencing and telephone conversations, hearing aid devices, voice-enabled assistants and more. Such term encompasses a wide variety of tasks and absolves to at least as many tasks as are the potential applications: voice activity detection and keyword spotting, denoising, dereverberation, diarization and so on, each performing an essential pre-processing step for a particular downstream use-case.

The goal of this dissertation is to give an overview of front-end speech processing and present different contributions to this important line of research that address many practical problems. More in detail, here we focus especially on the use of deep learning techniques, often supported by classical signal processing techniques, to tackle the front-end tasks of multi-channel speech enhancement, channel selection, keyword spotting, speaker counting and diarization.

Emphasis is placed on low computational complexity and/or low-latency approaches as well as integration between different front-end components to achieve one particular goal e.g. voice activity detection together with speech separation to obtain diarization or the use of spatial features to improve speaker counting.

Regarding multi-channel speech enhancement we present a study on the use of learnable filterbanks for acoustic beamforming which can open up interesting future research directions towards low-latency applications.

We also address the channel selection problem and propose to formulate it as a learning to rank problem. Our proposed MicRank algorithm is lightweight and can achieve performance in some instances close to oracle selection techniques.

Low computational requirements are also the primary goal of our implicit acoustic echo cancellation framework, which allows for streamable robust keyword spotting and device-directed speech detection on edge devices. It is also one of the main focuses of our study on overlapped speech detection and speaker counting on real world meeting corpora. Regarding this latter, we show that spatial based features could boost considerably the performance and at the same time keep the computational cost contained.

Finally we present a work on speech separation guided diarization for telephone conversations, in which we place special attention on extreme low-latency use-cases. The results are promising in terms of recognition and diarization performance and open up exciting prospects for applications such as live captioning.

Contents

List of Acronyms

AEC acoustic echo cancellation. 4, 57

AHC agglomerative hierarchical clustering. 92

AM acoustic model. 28

ASR automatic speech recognition. 3, 10, 14, 23, 25, 26, 28–30, 39, 45–47, 61, 91, 109–111

BIC bayesian information criterion. 92

CD cepstral distortion. 26

CNN convolutional neural networks. 2

CSS continuous speech separation. 96, 97

CTS conversational telephone speech. 98

DDD device-directed speech detection. 4, 6, 45, 58, 61

DNN deep neural networks. 2, 3, 5, 6, 9–11, 15, 16, 30, 62, 92, 107, 108

DSP digital signal processing. 5, 25, 107

EV envelope variance. 26

FAR false accept rate. 54–56

FLOPs floating point operations. 54, 56, 58, 111, 112

FOV receptive field. 56

FRR false reject rate. 54–56

GEV generalized eigenvalue. 9

GLR generalized likelihood ratio. 92

GMM gaussian mixture model. 62

GPU graphics processing unit. 2

GSCv2 Google Speech Commands v2. 52

GSS guided source separation. 39, 41, 43, 108

HMM hidden Markov model. 62

iAEC implicit acoustic echo cancellation. 5, 6

KWS keyword spotting. 4, 6, 47, 52, 58, 61, 112

LPC linear predictive coding. 92

LSTM long short-term memory. 2, 62

LTR learning to rank. 6

MFCC Mel-spaced frequency cepstrum coefficients. 92

ML machine learning. 5

MLP multi-layer perceptron. 62

MOS mean opinion score. 29, 31

MVDR minimum variance distortion-less response. 9, 12, 107, 108

MWF multi-channel wiener filter. 9, 12, 107

NLMS normalized least mean squares. 57

OSD overlapped speech detection. 4, 6, 89

OSDC overlapped speech detection and counting. 6

PIT permutation invariant training. 3, 95

PLP perceptual linear predictive. 92

RIR room impulse response. 15, 25

ROVER recognizer output voting error reduction. 25

SA-WER speaker-attributed WER. 39, 42

SCM spatial covariance matrix. 9

SI-SDR signal-to-distortion ratio. 14, 16, 97, 111

SI-SDRi SI-SDR improvement. 98, 100

SIR signal-to-interferer ratio. 45, 51, 52

SSE speech separation and enhancement. 5, 6, 23, 109–111

SSep speech separation. 96

SSGD speech-separation guided diarization. 6, 96, 101, 104, 105

SSL self-supervised learning. 112

SSLR self-supervised learning representation. 110

STFT short-time Fourier transform. 9–11, 13, 24, 62, 70, 72, 107, 108

SVM support vector machine. 62

TCN temporal convolutional network. 98, 99

TDNN time-delay neural network. 94

TDOA time-delay of arrival. 25

TTS text-to speech. 6, 45, 46, 51, 108

VAD voice activity detection. 3, 4, 6, 61, 62, 89, 92, 95, 97–99

WA word accuracy. 30, 32, 33

WER word error rate. 2, 3, 7, 28–30, 32, 33, 36, 38

WLM wiener-like mask. 16

WPD weighted power minimization. 110

Chapter 1

Introduction

Our ability to perceive sound plays a vital role in communication, learning, and interaction with each other and with the world around us. For example, it is well known that hearing appears much earlier than vision in the development of the fetus. Vision develops much later and even newborns have very poor eyesight. Thus, in the very first weeks of our existence, from the sounds of our mother's voice to the myriad of sounds of the outside world, we mostly rely on our ability to hear to make sense of the world around us. At the same time, this very same ability to process and produce sound has shaped our evolution over the course of millions of years and led to the development of language, which is thus intrinsically tied to speech and hearing. Indeed, the capability to communicate in such a rich manner is arguably one of the most impressive feats of human beings. It enabled us to share our knowledge, and experiences, exchange ideas, and ultimately build communities, societies and cultures. And it continues to do so.

This is why the technical field of audio and, in particular, speech processing is so important. It can transform the way we communicate and connect with each other and, in fact, it has been this this way since the invention of the radio and the telephone. Nowadays, advanced technologies such as speech recognition, natural language processing and machine translation have the potential to overcome language barriers and enable more effective communication across different cultures and communities. Think of live translation, once only imagined in science fiction, is now a reality, although further progress is needed to make it more robust.

Crucially, audio and speech processing can also play a critical role in improving accessibility and inclusion for people with hearing impairments or people with disabilities/learning disorders. For example, by devising better algorithms for hearing aids, thus helping people with auditory challenges in situations where classical hearing aid technology fails: during concerts, dinners and, in general, multi-party conversations in noisy places. Even helping individuals who lost hearing altogether with reliable live-captioning algorithms is a feat within reach of the current technology and could be a reality in the next few years. Also voice-enabled smart assistant have a lot of potential for assistive technologies. They offer an intuitive way for human-to-machine interaction, more accessible for many elderly people or people with disabilities, that cannot type for example or are not fully self-sufficient. It has been found, for example, that they can improve the quality of life of people with dementia or Alzheimer and even help with detecting early signs of cognitive impairment [1].

As said, many of these things 20 years ago belonged mainly to the real of science fiction. This drastic advancement is mainly due to three factors that are intimately related: the adoption of Deep Learning (DL) techniques, the greater availability of data and the progress in parallel-computation hardware e.g. especially in graphics processing unit (GPU). Since the turning of the millennium, with the rapid growth of the World Wide Web, massive amounts of data, in all forms: textual, audio-visual etc. has been more and more within reach. This led researchers to rediscover the use of deep neural networks (DNN) based machine learning techniques. The basis for Deep Learning are more than half a century old: the backpropagation algorithm was developed in the 60s and later mathematically formalized in the 70s by Rumelhart et al. [2]. The 80s saw the foundations of the convolutional neural networks (CNN) laid by Fukushima [3], while the 90s gave birth to long short-term memory (LSTM) networks [4], and saw the successful applications of CNN to practical problems such as handwritten recognition for signature verification [5].

The AlexNet [6] paper, now already 10 years old, is often pointed out as the turning point. Aside from winning by a large margin the ImageNet challenge it introduced several different novelties (e.g. the ReLU activation), and it was one of the first DNN-based techniques to use the full power of DL embarrassingly parallelism through GPU hardware and thus scale to large amounts of data.

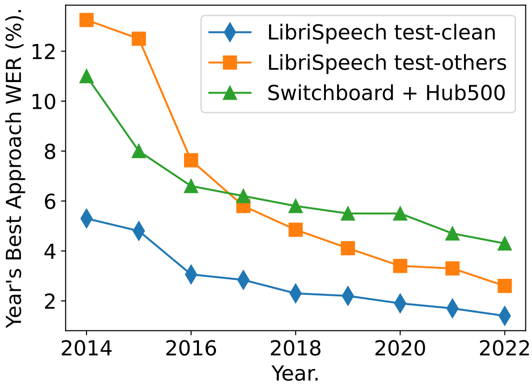


Figure 1.1: word error rate (WER) as obtained by the best system each year, from 2014 to 2022, on LibriSpeech [7] and Switchboard Hub5'00 datasets [8].

Since then DNN have been widely adopted and now are, ultimately, the de-facto standard approach to many speech processing problems. For example, regarding automatic speech recognition (ASR), a drastic reduction in the WER in most popular benchmark datasets can be observed from 2014 on-wards as we plotted in Figure 1.1. This is also reflected by the larger adoption of voice-enabled assistants, whose use has become mainstream now.

Other key developments in the field of deep learning with large impact in the audio field have been, to name a few, the invention of the Transducer [9], connectionist temporal classification loss [10], Transformer architecture [11] and permutation invariant training (PIT) to name a few. Another very recent major milestones is the development of effective self-supervised training strategies for in speech separation [12], for general speech representations [13–15], or general sound representation [16]. These latter have and are transforming the field as they require no supervision during training and can thus leverage extremely large amount of data during training. They are also flexible and can be adapted successfully for many downstream tasks [17]: e.g. ASR, diarization, emotion classification, speech translation et cetera.

To conclude, the extremely rapid progress of this field, is what ultimately makes it so stimulating to work with and try to be part of the audio and speech processing community. Every year it is easy to be amazed by some novel approach/algorithm with lots of potential.

This said, there are still so many challenges to be overcome, as we will also see in the rest of this manuscript. Deep learning methods are very powerful, but are data hungry thus raising both environmental [18, 19] and privacy concerns. They could be prone to overfitting a particular domain or scenario and can fail spectacularly when the domain is changed even slightly, in an imperceptible way (e.g. speech enhancement on top of ASR [20, 21], or adversarial attacks [22, 23]). All these phenomena are in stark contrast with the remarkable capability of humans to adapt quickly to new settings, contexts and acoustic conditions. Despite tabloids flamboyant claims, we are still quite far from developing speech processing techniques capable of tackling “in-the-wild” conversations and as flexible and reliable as our brain/auditory system.

1.1 Front-End Speech Processing

Audio front-end processing plays a critical role in all areas of speech and audio processing and will be the main subject of this thesis. As the name implies, it concerns with the pre-processing of the audio signal for the downstream applications. This term encompasses a wide-variety of techniques and tasks, aimed at, for example, improving the quality and reliability of speech and audio signals or discarding non-relevant audio portions e.g. via voice activity detection (VAD), the task of detecting speech segments. In fact, to process the speech signal, arguably you need to detect first where it is present in a given audio stream. It can involve one or, if available, more microphone channels or even more devices. In the latter case front-end processing could be used to fuse information across the different microphone channels e.g. as in spatial filtering or beamforming, in order, for example, to suppress unwanted noise.

In a broad sense, it can be said that it fulfill a function similar to the peripheral auditory system and primary auditory cortex: localizing sound sources, separating them in multiple independent streams and extract the most useful cues and necessary information from the "raw" sound vibrations, captured by our eardrums.

1.1.1 Taxonomy

Due to its broad scope, it is difficult to derive an exact taxonomy of front-end processing as the different tasks often overlap, e.g. historically VAD has been necessary to perform beamforming or acoustic echo cancellation (AEC).

In this manuscript, we divide front-end processing into three main categories, defined ultimately by the nature of the problem each task addresses:

- Speech Separation and Enhancement (SSE).
- Sound Localization [24].
- Audio and Speech Segmentation/Detection.

With SSE, following [25], we denote the front-end tasks of speech separation and enhancement. Enhancement is assumed here in the broader term, where it can possibly include also dereverberation on top of denoising. These tasks can be informed or blind and in the informed case, they are usually referred to as target speaker extraction [26]. In this dissertation we will present two works on such sub-field of front-end processing, one on joint enhancement and separation in the multi-channel case (Chapter 2) and another on monaural speech separation for diarization purposes (Chapter 6). In Chapter 3 we will study instead the channel selection problem. This problem too could be framed in an SSE sense, as it can be considered a limit case of spatial filtering with the filter being a one-hot vector.

Audio and speech segmentation/detection instead deal with the problem of sequence labeling. This term encompasses front-end tasks such as VAD, overlapped speech detection (OSD), speaker counting, keyword spotting (KWS), device-directed speech detection (DDD) and speaker diarization. In this dissertation a work on joint VAD+OSD and speaker counting will be presented in Chapter 5, one on diarization in Chapter 6 and finally we will also deal with on-device KWS and DDD in Chapter 4.

Another obvious sub-distinction we can make, within each of these three main categories is between monaural and multi-channel approaches. These latter are, in principle, more robust as they can exploit more information and the spatial diversity offered by the position of each microphone/device. In this work, multi-channel techniques are explored in Chapters 2, 3 and 5.

Finally, we could sub-divide further each front-end task/algorithm according to the mathematical formulation and framework adopted to tackle and describe the problem. Such distinction is often blurred but in general we can discriminate between “pure” digital signal processing (DSP) approaches, classical machine learning (ML) approaches and more recent approaches based on Deep Learning and DNN. It is not uncommon for the two latter approaches to always integrate DSP-derived “know-how”: e.g. for beamforming as it will be explained later in Chapter 2.

1.2 Scope and Organization of this Thesis

The goal of this thesis is to present different aspects of front-end processing, focusing in particular on speech applications, along with some contributions made in this field during this PhD journey. In the pages that follow, we will propose different novel algorithms in the areas of speech segmentation and speech separation and enhancement (SSE), highlight their advantages, shortcomings and also the potential applications. Our hope is that the reader will get an overview of the current state of research in many front-end speech processing areas, and maybe also be inspired with some new ideas and future research directions.

The works in this dissertations share two trends. The first is that we strive to focus on approaches that have low computational requirements and/or are suitable for low-latency streaming applications. This direction is often neglected in many studies but is crucial for practical applications, more so for front-end tasks, as they are usually required to run on edge devices. The second trend is integration between different front-end areas e.g. neural localization to aid in the task of speaker counting, or combining voice-activity detection and diarization to perform diarization. The meta-idea of integration between different tasks and components will be discussed further at the end of this work, but it is a recent general trend in speech processing which has been helpful in the design of more robust systems.

This dissertation focuses heavily on the practical aspect, and in each Chapter we explain the potential applications of the proposed frameworks and algorithms. But, at the same time, we place equal emphasis to the methodological aspect. Designing efficient and/or low-latency front-end solutions with state-of-the-art performance requires a lot of effort in the development of new methods. These include the development of completely new frameworks such as our implicit acoustic echo cancellation (iAEC) framework as well as the design of novel, efficient DNN architectures (such as the Transformer-based one with the *cat-pool* operation in Chapter 5). Or, again, for example, the use of approaches motivated by classical digital signal processing, such as spatial features, which can be used to boost the performance of speaker counting classifiers.

This work is organized as follows:

Chapter 2: we give a more in depth overview of SSE, focusing on multi-channel techniques and present a work on interpretable DNN-based beamforming in a learnt basis. This work shows that, by using learned filterbanks it is possible to surpass in some instances even oracle-based classical approaches.

Chapter 3: we introduce the channel selection problem and outline the main approaches. We then proceed to present MicRank, a framework we developed in which channel selection is framed as a learning to rank (LTR) problem. In detail we explore different LTR strategies and perform extensive experimental analysis on CHiME-6 and a purposely developed synthetic dataset. Results are promising and we show that such approach on single-talker data considerably improve over previous selection techniques and reach performance comparable and, in some instances better, than oracle signal-based measures. As an additional contribution, we also report an analysis over the use of signal-based channel selection in conjunction with speech separation, in the context of the recent CHiME-7 DASR Challenge.

Chapter 4: we present more in depth the two front-end tasks of on-device KWS and DDD. Our work on iAEC is then presented, we devise a novel framework to address in an efficient and effective manner a particular problem of human-machine interaction due to the user voice overlapping with the device text-to speech (TTS) response. Results show that our approach can obtain performances comparable to other state-of-the-art approaches but with more than 100 times less compute.

Chapter 5: we give an in-depth historical overview of VAD, OSD and speaker counting and then introduce our proposed overlapped speech detection and counting (OSDC) framework of which encompasses all these three tasks. We then proceed to study how supervised deep-learning methods can be used to tackle these tasks, focusing on real-world distant meeting scenarios with multiple microphones and on lightweight algorithms. We show that by using additional spatial features the performance can be increased considerably (even surpassing ensemble methods) at a modest increase in computational requirements. We also propose two novel carefully designed DNN architectures which achieve state-of-the-art performance while keeping the computational requirement low enough for on-device deployment.

Chapter 6: we present a work on speech-separation guided diarization (SSGD), focused on telephone conversations and low-latency. We carry extensive experiments with two state-of-the-art speech separation algorithms on CALLHOME [27] and Fisher [28]. A novel, efficient leakage removal algorithm is devised which is shown to drastically reduce false alarms due to single speaker segments. Results show that our proposed SSGD approach is an intriguing direction: it allows to get diarization "for free" on top of speech separation which is competitive with the state-of-the-art and, in the online case, has two orders of magnitude less latency. It also allows for decreasing the WER of the recognizer output thanks to separation, with results close to the oracle.

Chapter 7: we draw conclusions and outline possible future work directions as well as perspectives over the coming years regarding some of the challenges that we need to address.

Chapter 2

Speech Separation and Enhancement

Context

The work presented in this Chapter was presented at ICASSP 2022 [29] and was done together with Manuel Pariente from Université de Lorraine and Francois Grondin from Université de Sherbrooke. The idea actually came towards the end of 2019. After me and Manuel Pariente finished our work on learnable analytic filterbanks for monaural speech separation [30], this seemed a natural extension. Two years afterwards, also thanks to the, then just added, Pytorch complex numbers support, we decided that the times were ripe to give this idea a go.

2.1 Multi-Channel Enhancement Techniques

Most current deep learning based beamforming (also called *neural beamforming*) techniques can be divided into two main categories: *hybrid* [31–41] and *fully neural* [42–46].

Hybrid techniques couple DNN with established beamforming methods such as minimum variance distortion-less response (MVDR) [47], multi-channel wiener filter (MWF) or generalized eigenvalue (GEV) [48] solutions. Usually they employ a DNN to estimate the spatial covariance matrix (SCM) via a time-frequency mask [31, 34–39] in the magnitude short-time Fourier transform (STFT) domain. Another approach [40] is to use the DNN model to estimate the target and interferer time domain signals and subsequently derive the SCMs. In both cases the DNN is usually a monaural model and the mask is estimated on one microphone channel used as a reference. Additional spatial features are sometimes used to improve the masks estimation [32, 41]. As they rely on SCM estimation to derive the beamforming solution, hybrid neural beamformers performance is greatly affected by the frame size used to estimate the SCMs of the target and interferer/noise signals.

On the other hand, fully neural models employ a DNN to directly estimate the beamforming filters [42, 43] or the time domain target signal directly [44, 49, 50] or, via complex spectral mapping, the target STFT [51, 52]. Being fully data-driven these methods are less sensitive to the frame size of the beamforming filters. For example FasNet [42] is able to reach comparable or superior performance with respect to conventional oracle beamformers for low latency applications with remarkably smaller frame size and latency. This is aligned with results in monaural source separation, where fully learned representations have been shown to surpass the STFT in both clean [53] and noisy conditions [30] especially for short windows [54]. They also have a potential for being computationally lighter than hybrid approaches, as MVDR and MWF require expensive matrix inversion operations. However, fully neural models are also arguably “less interpretable" and are prone to introducing non-linear distortion compared to conventional beamformers. FasNet [42] is a notable exception as it estimates linear spatial filters for filter-and-sum beamforming, thus enabling to e.g. visualize the beam-patterns. This however is not possible for other methods [44, 49, 50] as the multi-channel processing is done inside the DNN.

Still, conventional beamformers and especially MVDR are preferred in actual applications, in particular ones involving deep learning based systems downstream (e.g. for ASR), as the distortion-free constraint is crucial for not introducing artifacts that would cause a domain mismatch [21]. This is why hybrid DNN approaches are still very relevant as also recent works suggest [55–58].

2.2 Acoustic Beamforming with Learned Filterbanks

Given these premises, it would be interesting to attempt to bridge the gap between these two paradigms and study conventional beamforming with fully learned filterbanks, as these latter also achieved promising results in monaural source separation [30, 53, 54]. We propose to train a hybrid neural beamformers where the DNN is used to estimate the SCMs via a mask. However unlike previous works [31–41] we learn the analysis and synthesis filterbanks in place of the STFT along with the mask-estimation DNN using time-domain losses. We consider for this study fully unconstrained linear filterbanks as used in [53] and the recently proposed learnable analytic filterbanks [30] which allow for magnitude shift invariance, an especially desirable property in this case.

We will outline now the signal model and framework which will be used in the rest of this Chapter.

Given an array of M microphones we can denote with

$$\boldsymbol{y}(t) = [y_1(t), y_2(t), \ldots, y_M(t)]^T \,,$$

the matrix of the time-domain signals at each microphone, with t being the sample index. We consider here a situation where $\boldsymbol{y}(t)$ is comprised of two terms:

$$\boldsymbol{y}(t) = \boldsymbol{x}(t) + \boldsymbol{\nu}(t), \tag{2.1}$$

with $\boldsymbol{x}(t)$ the matrix of desired source signals and $\boldsymbol{\nu}(t)$ the matrix of interfering source signals at the microphones. Our goal here is recovering the desired signal $x_r(t)$ at an arbitrarily chosen reference microphone $1 \leq r \leq M$ by suppressing the interferer. This implies that, in this study, the target is a reverberated source signal and joint enhancement and dereverberation is left for future work. Accordingly, the target signal at reference microphone r is given by $x_r(t) = \sum_{\tau=1}^{L_h} h_r(\tau) x^a(t-\tau)$, where $x^a(t)$ is the dry desired source signal and $h_r(\tau)$ is the impulse response of length L_h characterizing the acoustic propagation of the desired source signal to the reference microphone at time lag τ. Recovering of $x_r(t)$ can be achieved by conventional spatial filtering techniques if an estimate of the target signal and the interferer SCMs can be produced.

Hereafter we follow a simple hybrid neural beamforming framework, illustrated in Figure 2.2, where such estimates are produced by a monaural mask estimation DNN $\mathcal{F}(\cdot, \boldsymbol{\theta})$ with $\boldsymbol{\theta}$ trainable parameters. An STFT analysis filterbank $\boldsymbol{\phi}_n(t)$ is used to extract the time-frequency representation for every m-th microphone input signal, obtaining a third order tensor:

$$Y_m(n,k) = \sum_{t=1}^{L} y_m(t+kH)\phi_n(t), \;\; n \in [1, \ldots, N] \,, \tag{2.2}$$

where $\{\phi_n(t)\}_{n=[1,\ldots N]}$ are the N STFT analysis filters each of size $L = N$ and H is the hop-size or stride factor. Consequently n and k denote respectively the frequency bin and frame indexes.

The mask-estimation DNN has, as input features, this complex STFT representation (real and imaginary part) at a chosen reference channel r and outputs a mask $m(n,k)$ for the target signal in the magnitude STFT domain (i.e. with shared values between real and imaginary parts):

$$m(n,k) = \sigma(\mathcal{F}(Y_{m=r}(n,k), \boldsymbol{\theta})) \tag{2.3}$$

where $\sigma(\cdot)$ denotes the sigmoid activation. The interferer signal mask is obtained simply as $1 - m(n,k)$. We found this configuration to work the best in our experiments rather than outputting two distinct masks and/or using a different activation (e.g. softmax). More in detail, this configuration led to more stable training with the dataset used in our experiments, while the use of two distinct masks often led to ill-conditioned SCMs especially when the activation was unbounded (e.g. ReLU).

These masks are then used to compute the frame-wise SCMs of target and interferer respectively:

$$\begin{aligned}\boldsymbol{R}_x(n,k) &= \boldsymbol{Y}(n,k)m(n,k)\boldsymbol{Y}(n,k)^H,\\ \boldsymbol{R}_\nu(n,k) &= \boldsymbol{Y}(n,k)(1-m(n,k))\boldsymbol{Y}(n,k)^H,\end{aligned} \tag{2.4}$$

where H denotes the Hermitian transpose and both target and interferer SCMs are 4-th order tensors $\in \mathbb{C}^{M\times M\times N\times K}$, where K is the number of frames. In this study, for simplicity, we consider non-causal systems. In this instance, following previous works [31, 34–40], the overall SCM can be computed by simply averaging the frame-wise SCM over the whole input mixture segment: $\boldsymbol{R}_\rho(n) = \frac{1}{K}\sum_k \boldsymbol{R}_\rho(n,k)$ for both target $\rho = x$ and interferer $\rho = n$. In addition to non-causality, this averaging operation requires that the transfer functions of the target source and interferer do not change in the time-frame over which the averaging is performed i.e., for the target, h_r is assumed stationary.

From such estimated SCMs different beamforming solutions can be computed. In this study we consider MVDR and MWF.

Regarding MVDR, we use the formulation from [59] and estimate the spatial filter as

$$\boldsymbol{w}_m^{MVDR}(n) = \frac{\boldsymbol{R}_\nu^{-1}(n)\boldsymbol{R}_x(n)\boldsymbol{u_m}}{tr\left\{\boldsymbol{R}_\nu^{-1}(n)\boldsymbol{R}_x(n)\right\}}, \tag{2.5}$$

where $tr\{\cdot\}$ denotes the trace operator and $\boldsymbol{u_m}$ is an one-hot column vector for which the m-th term is 1, and all others are 0. Regarding MWF instead we simply compute the filter coefficients as:

$$\boldsymbol{w}_m^{MWF}(n) = \left(\frac{1}{\boldsymbol{R}_x(n)+\boldsymbol{R}_\nu(n)}\boldsymbol{R}_x(n)\right)\boldsymbol{u_m}, \tag{2.6}$$

and the beamformed signal is obtained as

$$\tilde{X}(n,k) = \boldsymbol{w}_m(n)^H\boldsymbol{Y}(n,k), \tag{2.7}$$

which is finally brought back to time-domain via a *synthesis* inverse-STFT (iSTFT) filterbank $\boldsymbol{\psi}_n(t)$ filterbank with N synthesis filters $\{\boldsymbol{\psi}_n(t)\}_{n=1,\dots N}$ of, again, length $L = N$ each:

$$\tilde{x}(t) = \sum_{k=1}^{K}\sum_{n=1}^{N}\tilde{X}(n,k)\psi_n(t-kH). \tag{2.8}$$

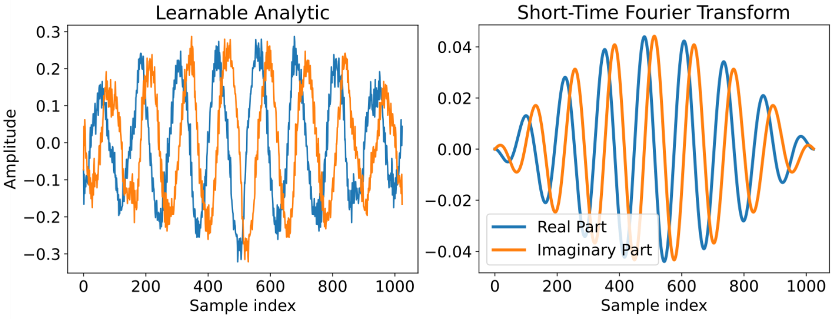


Figure 2.1: Left: an example of a learnable analytic filter, taken from a trained model we used in our experiments. The filter belongs to the analysis filterbank and has a 1024 kernel size. Right: a visually similar filter, as found in a STFT filterbank with the same kernel size.

2.2.1 Learnable Analysis and Synthesis Filterbanks

In this work we propose to replace the STFT and iSTFT filterbanks with learnable linear filterbanks and perform spatial filtering in a learned linear basis. These filterbanks are learnt end-to-end jointly along with the mask-estimation DNN $\mathcal{F}(\cdot, \boldsymbol{\theta})$ as the gradient can be back-propagated from a time-domain loss also to the analysis and synthesis filterbanks.

We consider here two types of filterbanks: *free* and *analytic* [30] ($\mathcal{A}$) along with the STFT. In free filterbanks both analysis and synthesis parameters are unconstrained as in [53] with N fully learnable filters.

On the other hand, learnable analytic filterbanks have only half of the filters fully learnable. For example, regarding the analysis filterbank $\{\boldsymbol{\phi}_n(t)\}_{n=1,\dots N}$, we consider the first $N/2$ filters as the real and fully learnable part while, the corresponding imaginary part is obtained from its real counterpart via the Hilbert transform $\mathcal{H}(\cdot)$:

$$\boldsymbol{\phi}_n^{\mathcal{A}}(t) = \boldsymbol{\phi}_n(t) + j\mathcal{H}(\boldsymbol{\phi}_n(t)). \tag{2.9}$$

The same is true for the synthesis filterbank $\{\boldsymbol{\psi}_n(t)\}_{n=1,\dots N}$. For implementation purposes the analytic filterbanks real and imaginary parts are treated separately as N real filters and the whole filterbank is implemented as a 1D convolutional layer[1]. An example is reported in Figure 2.1. The filter is taken from a 1024 samples kernel size analysis filterbank belonging to one of the models used in our experiments in Section 2.4. Because of this coupling, the modulus of a signal convolved with these learnt filters is invariant to small shifts in time domain, a property shared with the STFT. This property is cru-

[1] see github.com/asteroid-team/asteroid-filterbanks

cial for the estimation of the SCMs in Equation 2.4, as the target signal mask is estimated on the reference channel and applied across all microphones.

The use of fully learnable filterbanks in place of the STFT poses some problems regarding the derivation of the SCMs. An implicit assumption for Eq. 2.4 is that the analysis filterbank used is approximately orthogonal. This condition is commonly referred to as the “narrow-band approximation" and can be satisfied by the STFT because of its approximate orthogonality [60]. At least under some assumptions [60], mainly related to the maximum delay in the relative transfer function of the array and the length of the analysis window.

Without this assumption, the target and interferer SCMs cannot be reduced to $M \times M$ matrices as in Eq. 2.4, as with no orthogonality of the basis, one must take into account also “inter-frequency" terms. This leads to a block matrix SCM for each frame k that can be partioned as an $N \times N$ block matrix (modeling the inter-frequency interactions) with the $(i,\ j)$-th block being a $M \times M$ matrix (modeling the inter-microphone interactions). This increases significantly the computational requirements as e.g. inversion of the full SCM leads to a complexity of $\mathcal{O}(N^3M^3)$ versus $\mathcal{O}(NM^3)$ for a diagonal block SCM.

A straightforward, naive, but efficient approach, is to disregard the contribution of the “inter-frequency" interactions in the SCM derivation also for the learned filterbanks. Since the filterbanks are learnt jointly with the rest of the model by minimizing a particular loss objective (e.g signal-to-distortion ratio (SI-SDR)) it can be assumed that the analysis filterbank will learn an approximately orthogonal basis. We adopt in this work this rather strong assumption and provide some empirical evidence for this in Section 2.4.

2.3 Experimental Setup

2.3.1 Datasets

Crucially, most neural beamforming studies, being targeted mainly towards back-end tasks such as ASR, perform their experiments using 16 kHz signals. Such sampling rate however is sub-optimal for applications aimed towards human listening. For this reason, we use in our experiments the recently available First Clarity Enhancement Challenge dataset [61] which, being geared towards hearing aid development, is sampled higher at 44.1 kHz.

Clarity Challenge Dataset

We use here the training and development subsets from the Clarity Challenge comprised of, respectively, 6k ($\sim$ 10 h) and 2.5k ($\sim$ 4 h) multi-channel simulated mixtures. Each simulated mixture consists in a target speaker and an interferer signal which can be either another competing speaker or a localized

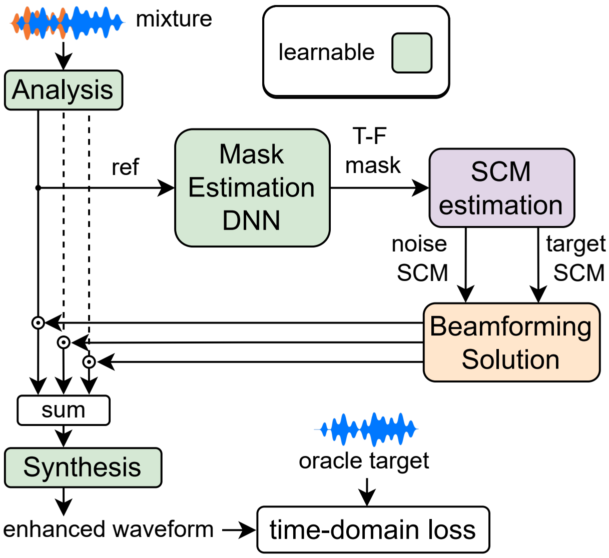


Figure 2.2: Framework overview. The gradient is back-propagated from waveform domain. This allows to learn the analysis and synthesis filterbanks along with the mask-estimation DNN.

noise source. By dataset construction, each mixture is composed in such a way that the interferer signal always starts 2 seconds before the target signal. To make the task more challenging, in this work we only use 1 second of such "preroll". Spatialization is performed using synthetic room impulse response (RIR) by simulating a randomized room with uniformingly sampled receiver, target and interferer locations, each constrained to be at least 1 m apart from the others. The RIR reverberation time at 60 dB (RT60) has a log-normal distribution with a mean of 0.3 s and a standard deviation of 0.13 s. The Raven toolkit [62] is used to perform such simulation. An array with a behind-the-ear hearing aid topology is employed with 3 microphones per ear. On each ear, microphones are spaced approximately 7.6 mm (front, mid, rear) from one to another. We consider the task of recovering the reverberant target signal at one reference microphone without considering the head related impulse response. In this dataset, the SI-SDR at the array between the target and interferer signals has a -30 to 10 dB range with a skewed gaussian distribution centered around 1 dB. In this work, we report results using the development and use a 90/10 training set split for the purpose of training and validation respectively.

2.3.2 Architecture and Training Details

In our experiments we employ ConvTasnet [53] separator as the mask-estimation DNN in Figure 2.2. We train the whole system comprised of analysis, synthesis mask-estimation DNN and beamforming solution in an end-to-end fashion using negated time-domain SI-SDR [63] as the loss function. Adam [64] is used for optimization along with gradient clipping for gradients exceeding an $\mathcal{L}_2$ norm of 5. We tune learning rate and weight decay for each experiment and train each model for a maximum of 100 epochs with early stopping if no improvement is seen in the last 10 epochs on the validation set. We halve the learning rate if no improvement is seen in the last 5 epochs. During training we randomly choose the reference channel from the 6 available while in testing and validation we always use the first left microphone as the reference.

2.4 Experimental Results

In our experiments we consider, as an upper bound, MVDR and MWF beamformers with oracle wiener-like mask (WLM) in STFT domain. We use as performance metrics SI-SDR improvement (SI-SDRi) and Signal-to-Distortion Ratio [65] improvement (SDRi). The SI-SDR and SDR values for no enhancement are respectively 1.537 dB and 1.144 dB. Note that these metric emphasize the contribution of the lower part of the spectrum as this also carries most of the energy for speech signals. Better objective metrics that allow for a more fair assessment of speech enhancement for signals with such high sampling frequency (44.1 kHz and above) are still a matter of ongoing research at the time of this writing. As here we are merely comparing the proposed method with oracle beamforming solutions and ones based on STFT we argue that the use of SI-SDRi and SDRi can still be considered acceptable.

In Figure 2.3a we report the SI-SDRi versus the length of the analysis and synthesis filters (*kernel size*) for different configurations. The number of filters is kept equal to the kernel size, and the stride half of that.

We can see that for both the STFT-based ($STFT$) models and oracle ($oWLM$) masks, performance improves as the kernel size increases. This is expected as a bigger kernel allows for more accurate SCMs estimation. Both *free* and $\mathcal{A}$ learned filterbanks outperform oracle WLM mask for small kernels. Only for MVDR, this is true also for all kernel sizes considered. Interestingly, learned filterbanks seem to have opposing trends regarding MVDR and MWF in function of the kernel size. For MWF performance decreases as the kernel increases. This may be due to the fact that learning filterbanks with large kernel sizes is inherently more difficult and leads to more “noisy training” as far as MWF is considered. On the contrary, the MVDR distortion-less constraint could

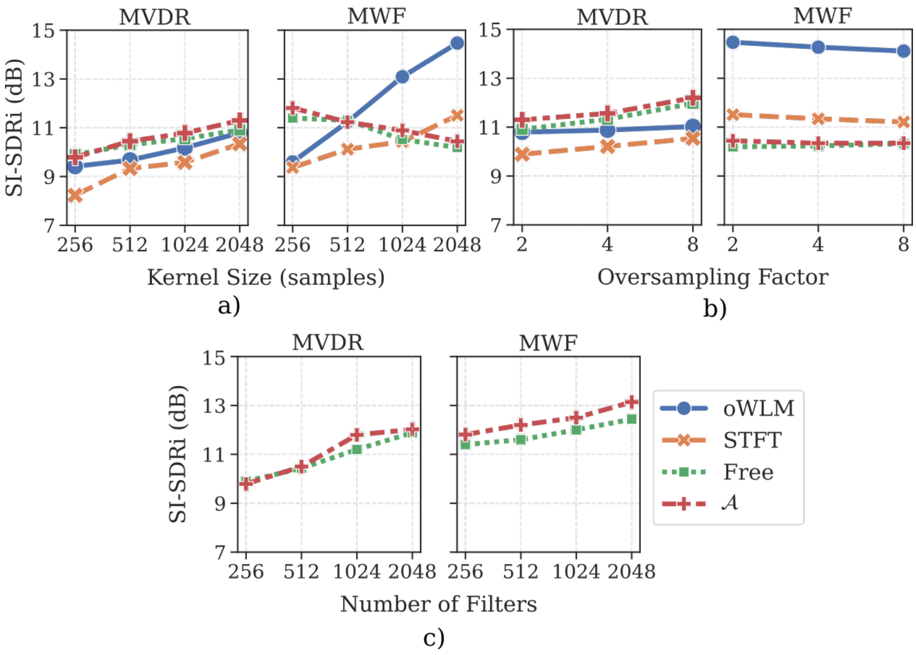


Figure 2.3: Performance for different MVDR and MWF configurations: oracle (oWLM) and learned models with different filterbanks (STFT, Free and $\mathcal{A}$).
a) SI-SDRi versus kernel size. The number of filters is kept equal to kernel size and stride to half. *b)* SI-SDRi versus oversampling factor. The kernel size and number of filters is kept to 2048. *c)* SI-SDRi versus number of filters for learnable filterbanks. The kernel size and stride are kept fixed at 256 and 128 respectively.

mitigate this issue.

In Figure 2.3b we study how SI-SDRi changes by increasing the oversampling factor i.e. decreasing the stride while keeping fixed the kernel size. Here we fix the kernel size and number of filters to 2048 and vary the oversampling factor N/H by 2, from 2 (same as in Figure 2.3a) to 8.

Regarding MVDR, for both STFT-based systems and *oWLM* performance improves with higher oversampling but at a slower pace compared to what has been observed by increasing kernel size. Regarding MWF, performance decreases slightly for STFT and *oWLM* while is almost constant for the models with learned filterbanks.

In Figure 2.3c we explore the effect of increasing the number of filters for learned filterbanks with fixed kernel size and stride of respectively 256 and 128 samples. Such strategy is, in fact, one of the key factors which allows current monaural source separation algorithms to achieve such impressive performance [53, 54].

For both beamforming solutions increasing the number of filters and thus forming an over-complete dictionary, improves significantly the performance. By comparing with Figure 2.3a, we can see that adding filters has a stronger effect with respect to expanding the kernel size. This suggests that beamforming with learned filterbanks may be particularly suited for low-latency applications as the kernel size can be kept low to suit the latency constraints, while the number of filters increased with no penalties in terms of latency.

In Table 2.1 we compare the best systems from previous experiments (Figure 2.3) in terms of both SI-SDRi and SDRi. As a term of comparison we also add iFasNet [44], a state-of-the art fully neural beamformer architecture. For this model we use the exact same configuration as in [44]: as the sampling rate here is 44.1 kHz here, iFasNet has more parameters compared to the original one due to increased window length.

The proposed approach is competitive with the current state-of-the-art. Among the non-oracle algorithms, MWF with learned filterbanks obtains the highest figures with the one based on analytic filterbank being the best. This latter consistently surpasses even the best oracle MVDR result.

In Figure 2.4, we report, at each training epoch, the Mean Absolute Cosine Similarity (MACS) for the analysis filterbanks of MVDR and MWF models with learned filterbanks. In detail, to measure the orthogonality of the learned filterbank, we compute the cosine similarity over each unique pair of filters, take the absolute value and take the average over the total number of unique pairs. We can see that the MACS value decreases during the training, indicating that the analysis filterbank gets more orthogonal as training progresses. This partly confirms the hypothesis made at the end of Section 2.2.1. On the other hand, the learned filterbanks converge, at best, to a MACS value of 0.013 which is

Method	N	L	H	SI-SDRi [dB]	SDRi [dB]	Params
oWLM-MVDR	2048	2048	256	11.023	12.410	-
oWLM-MWF	2048	2048	1024	14.733	15.551	-
STFT-MVDR	2048	2048	256	10.321	12.025	5.2M
STFT-MWF	2048	2048	1024	11.556	12.667	5.2M
free-MVDR	2048	256	128	11.882	12.963	6.3M
free-MWF	2048	256	128	12.435	13.632	6.3M
$\mathcal{A}$-MVDR	2048	256	128	12.024	13.372	5.8M
$\mathcal{A}$-MWF	2048	256	128	**13.142**	**14.272**	5.8M
iFasNet [44]	-	-	-	9.896	10.342	4.4M

Table 2.1: Comparison of best performing models in terms of SI-SDRi and SDRi and number of parameters (*Params.*).

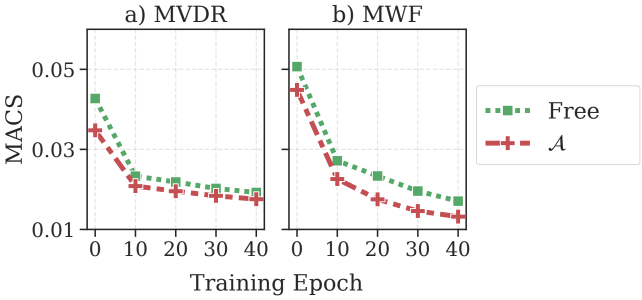


Figure 2.4: Mean Absolute Cosine Similarity (MACS) versus training epochs for learned filterbanks (Free and $\mathcal{A}$) MVDR and MWF models. All filterbanks have 1024 filters, 1024 kernel and 512 hop-size.

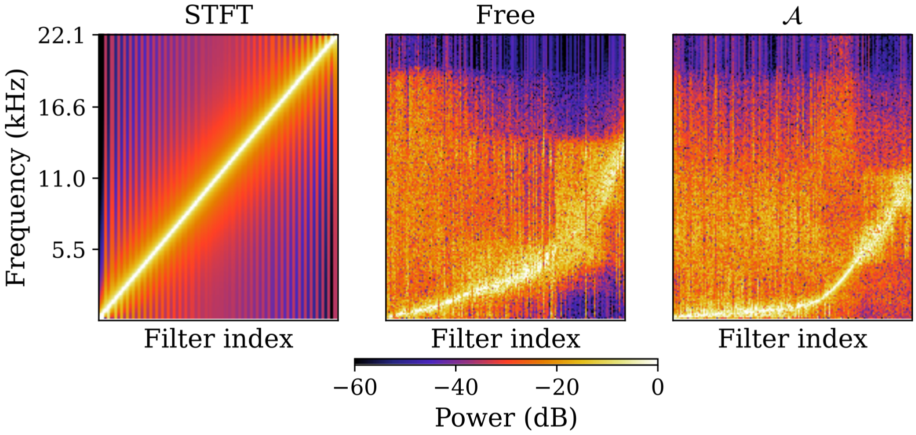


Figure 2.5: Frequency response of STFT, free and $\mathcal{A}$ filterbanks. All filterbanks have 2048 filters with 2048 samples kernel size. For visualization purposes, filters in learned filterbanks are sorted according to their center-band frequency.

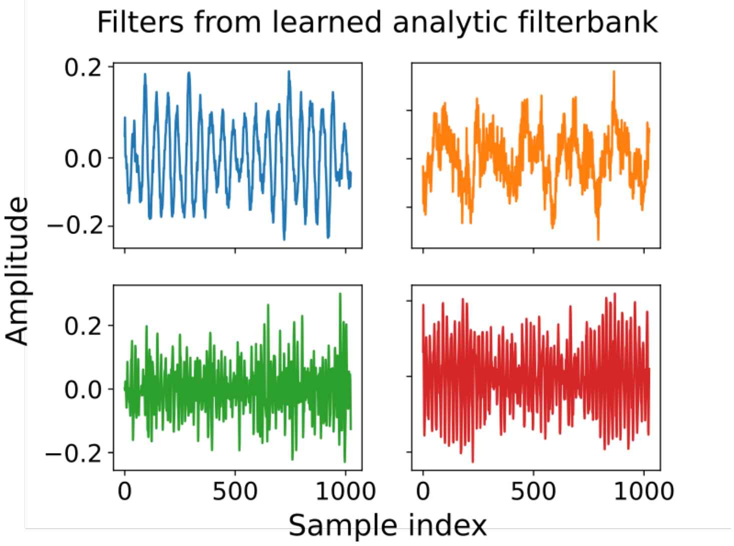


Figure 2.6: Some analytic learned filters (real-part only), taken from a trained model we used in our experiments. The filters belong to the analysis filterbank and consists of 1024 samples each.

more than one order of magnitude higher than 0.001, obtained for an STFT filterbank with same 1024 kernel size and number of filters. Future work could explore orthogonality constraints and their impact on performance.

In Figure 2.5 we illustrate the frequency response of STFT and the learned filterbanks under study. Both learned solutions tend to focus more on the lower part of the spectrum where most of speech energy is concentrated. In fact, for free and $\mathcal{A}$, less filters are localized in the higher end of the spectrum, following loosely an exponential trend which is less steep than Mel-scale. This is especially true for $\mathcal{A}$ as most of the filters have a center-band frequency in the sub 2 kHz region leading to an almost piece-wise linear trend. Free filters tend to have an higher frequency spread than analytic ones. Finally in Figure 2.6 we plot some learned filters from an analytic filterbank (same as the one in Figure 2.1). We can see that some filters exhibits some periodic structure, but this is not always the case, others appear more “noisy” and are more difficult to interpret as they have a wide-band frequency response.

2.5 Conclusions & Future Work

In this Chapter we investigated DNN-supported multi-channel speech enhancement with learned filterbank. We proposed a fully end-to-end hybrid neural beamforming framework, where a DNN is employed to estimate the SCMs used to derive conventional beamforming solutions such as MVDR and MWF. Differently from previous works, we consider the possibility to learn jointly with the DNN also the analysis and synthesis filterbanks instead of using the STFT and iSTFT. We carried an extensive experimental study comparing learned filterbanks with STFT investigating how performance changes with different kernel sizes, stride factors and number of filters. Two types of learned filterbanks have been considered: fully learned ones, which don't have any constraint, and analytic ones, which, by design, display shift invariance. We found that such proposed strategy of performing spatial filtering in a learned representation is particularly effective for the MVDR beamformer. In fact, in this case, we found learned filterbanks to consistently outperform STFT-based ones, even when oracle masks are employed. Regarding MWF, we found out that a gain over oracle masks is possible only for small kernel sizes. This suggests that future work could explore causal, low-latency applications. Among the two learned filterbanks considered, the analytic ones fare the best. This promising result suggests that it may be worth exploring additional inductive biases for learned filterbanks such as orthogonality constraints.

Chapter 3

Learning to Rank Microphone Channels

Context

MicRank [66] was presented at Interspeech 2021 and is a joint collaboration with Alessio Brutti and Marco Matassoni from Fondazione Bruno Kessler. The idea came in 2019 when we were together at JSALT 2019 working on CHiME-5 data, thanks to a seminar on learning to rank approaches within the JSALT workshop. A special thanks goes also to Maurizio Omologo who coordinated the JSALT workshop together with Mirco Ravanelli.

3.1 The Channel Selection Problem

In most scenarios, we can envision the presence of multiple heterogeneous recording devices. Consider for example a company meeting, as depicted in Figure 3.1, where at least some participants could have a smartphone or a laptop, or there could be far-field arrays, as in a teleconferencing room. In such situations, for the purpose of meeting transcription, we would like to exploit each device/microphone in order to minimize the transcription errors. This however is not a so easy feat: results on the recent CHiME-5 [67] and CHiME-6 [68] demonstrate that fully exploiting ad-hoc, possibly heterogeneous microphone networks is still an open issue.

As explained before, multiple audio streams could be used for beamforming, or in general multi-channel SSE in order to enhance each speaker signal and then feed it to the back-end task e.g. ASR. However, most of multi-channel SSE approaches [38, 69–71] are not designed for multi-device processing which, as said, is particularly challenging due to lack of precise synchronization between devices and the relative positions of these (they could be in different rooms for example). Lack of synchronization could lead to severely misaligned signals between different devices, for example, due to clock drift and packet losses, as

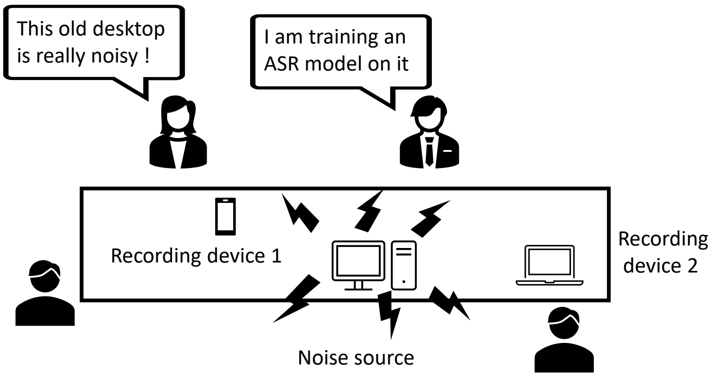

This old desktop is really noisy !
I am training an ASR model on it
Recording device 1
Recording device 2
Noise source

Figure 3.1: Application scenario: a meeting with multiple participants, some heterogeneous devices that can be used for transcriptions. Which is the best device/microphone for each speaker ? Is it always the closest one ?

observed in the CHiME-5 challenge dataset (see Figure 3.2). But also small misalignment (tens of milliseconds) is known to degrade performance. For example, STFT-domain beamforming techniques implicitly assume that the misalignment of the signals is much smaller than the STFT window length used. If this is not met, the STFT narrow-band approximation does not hold anymore and the beamforming results will be heavily degraded.

Nonetheless, during the years, also some multi-channel SSE methods purposely targeted towards ad-hoc microphone networks have been proposed [72–74]. The latter two [73, 74] assume perfect synchronization between devices while, [72] proposes a pipeline that includes re-alignment of the different audio streams prior to beamforming.

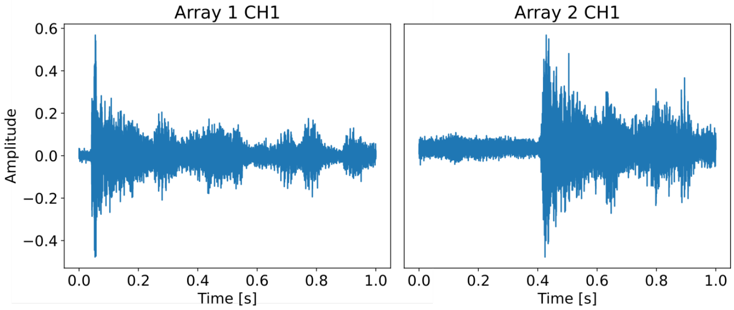


Figure 3.2: An example of signal-level misalignment between two array devices (but same microphone, CH1) on the CHiME-5 Challenge dataset [75].

An arguably simpler but still very intriguing approach is trying to select, for each speaker and each utterance, the best possible microphone that will lead to the lowest error (among all available microphones) for a considered back-end task. Such approach could be very effective too, as nowadays single-channel ASR models trained on large scale data such as Whisper [76] have become increasingly robust. In this way, since no combination is made, the synchronization issue disappears, and moreover, there could be a significant computational saving as channels are not combined anymore with expensive SSE algorithms. Or alternatively, one can also select a subset of the most useful channels, so that the computational requirements of SSE algorithms could be reduced significantly (e.g. as said in Chapter 2, most beamforming solutions have $\sim N^3$ complexity in the number of channels N). The saving is even more evident if ensembling back-end techniques are used directly over the multiple channels, e.g. recognizer output voting error reduction (ROVER) [77], as they are very computational demanding.

This channel selection problem has been widely investigated in the past, and the approaches proposed could be roughly be classified into four main categories:

- Signal-based hand-crafted features.
- Decoder-based measures.
- Posterior-based features.
- Data-driven methods.

We explain each in detail thereafter.

3.1.1 Signal-Based Hand-Crafted Features

As the name implies, these channel selection methods are based on hand-crafted features, carefully engineered according to DSP and acoustic principles, in order to be indicative of a channel quality.

Several of such methods have been proposed in the past [78–82]. These include measures such as estimated signal-to-noise ratio [79], ratio between late reverberation components and whole RIR [78] as well as more elaborate approaches [80–82]. These include also approaches based on localization, such as the one proposed by Kumatani et al. [80], that relies on cross-correlation between the available channels. However, this also requires the channels to be sample-synchronized, otherwise the time-delay of arrival (TDOA) could not be estimated easily, thus such approach could be difficult to employ across ad-hoc networks with heterogeneous devices. Wölfel [83] instead investigate the use of class separability, with a framework based on linear discriminant analysis they propose to learn from the features a within-class and a between-class matrix for linear regression of different target classes (e.g. acoustic sub-units). The channel that maximizes the class separability is assumed to be the best. Since these matrices are learned this method can be also considered one of the first data-driven methods for channel selection, however, it relies heavily on the use of hand-crafted features due to the use of a linear regression approach.

In the seminal work by Wolf and Nadeau [81], a very effective measure is proposed based on envelope variance (EV) and it is, to date, amongst the most effective blind channel selection methods. The core idea is that EV is able to model the reduced dynamic range in the speech intensity induced by reverberation quite effectively and, in [81], it is in fact shown to outperform even decoder-based measures. EV relies, as the name implies, on the variance of the mel-scaled filterbank energies after utterance-wise mean normalization in log-space (used to remove the short term effects as e.g. impulse response of the microphone). An example is reported in Figure 3.3, where the variance of each of the 40 Mel-subbands is plotted for an utterance from the CHiME-5 dataset and two different channels. We can notice that EV gives correctly the higher score to the best channel, the left one which appears less noisy and whose speech has clearly more energy (see the amplitude values on the waveforms).

A more recent work by Guerrero et al. [82] proposes to use the cepstral distortion (CD) both as an informed and a blind channel selection measure. It is roughly defined as the mean l_2 distance between a reference cepstrum and each channel cepstrum. In the blind case the reference cepstrum is taken from the average in the log-magnitude domain across all microphones. This is justified by the fact that it is assumed that such average in log-domain would be dominated by the closest microphone channel. However, we argue that such averaging operation could also lead to failure cases for this approach when the signals from the different microphones are severely misaligned.

Overall, the main advantage of all these signal-based methods is that they are inexpensive with respect to the other selection approaches, as they rely on extremely simple operations, which are also often optimized or have dedicated hardware (e.g. in the case of fast Fourier transform and cosine transforms for the cepstral coefficients). Moreover, they also don't require an extensive training set, but it suffices only some development/adaptation data for tuning some hyper-parameters such as the sub-band weights in EV.

3.1.2 Decoder-based Measures

Decoder-based channel selection methods were among the earliest one proposed along with signal-based ones. They rely, as the name implies, on measures extracted after the ASR decoding step.

For example Obuchi [84, 85] devised a channel selection method which per-

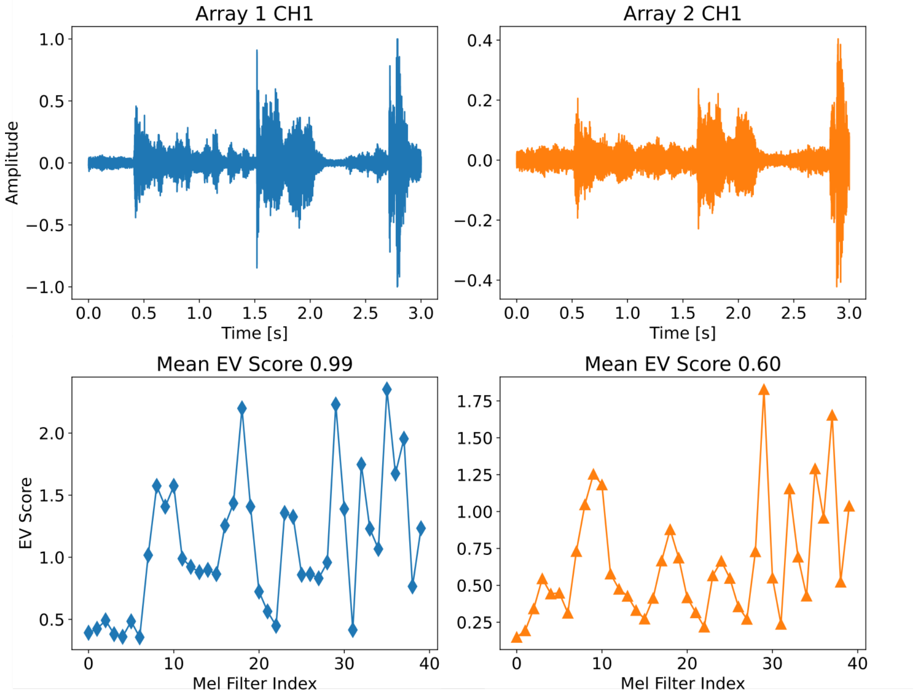


Figure 3.3: Envelope variance (EV) output on a segment from CHiME-5 for two different array devices, same microphone (CH1). We can see, from the waveform, that the right channel is more noisy. Accordingly, the EV values are, in fact, overall lower.

form channel selection by comparing the transcripts produced when the input to the ASR is normalized versus when it is not. The channel with the lowest distance e.g. in terms of WER is assumed the best one. This method however is impractical, especially with the ASR models we have nowadays, decoding two times all channels is prohibitively expensive. Moreover, it cannot be applied easily to modern DNN-based techniques, because these are very sensitive to input normalization. If a model is trained with or without normalization, then inference should be performed in the same way, otherwise performance could drop catastrophically.

Wolf and Nadeau [86] propose instead to use the likelihood ratios between the channels as a selection criterion:

$$\hat{C} = \arg\max_{m} \sum_{i}^{M} \frac{p(\mathbf{O}_m|\mathbf{w}_m)}{p(\mathbf{O}_m|\mathbf{w}_i)}$$

where $m = 1, \ldots, M$ and $i = 1, \ldots, M$ are the channel indexes. This is equivalent to selecting the channel that gives overall the highest confidence among all the other ones. The drawback of this method is that it is asymptotically quadratic in the number of channels if all the hypothesis $\mathbf{w}_i$ are different.

3.1.3 Posterior-based Measures

Posterior-based measures rely on the output posteriors of the acoustic model (AM) to perform selection. As such, they are quite computational demanding (but less than decoder-based measures) as the AM forward-pass has to be performed for each channel independently. In [87] a channel selection approach based on an entropy measure of the AM posterior probabilities is proposed and validated on the arduous CHiME-5 [75] dataset. The main idea is that an higher entropy measure and thus uncertainty in the AM predictions suggests a lower quality microphone channels. In their approach they propose to use a different AM than the one actually used for the transcription, trained only on clean data. The idea is that this model will be more sensible to acoustic degradation than the one trained with multi-condition training; thus leading to better channel selection. This has also the potential advantage that this clean AM could also be made smaller. However, it can be argued that there is also a drawbacks to this approach: the mismatch between the two AMs in the training data does not guarantee that the channel with lowest entropy for the clean AM is actually the best for the one trained with multi-condition.

3.1.4 Data-driven Methods

Recently some fully data-driven methods for channel selection have been proposed [66, 88, 89]. The first work in this direction is our proposed MicRank framework [66], which will be detailed in the next section. Another recent work [88], proposes to perform channel selection directly inside the ASR model, by using a sparsemax operator [90] (a sparsified variant of the popular softmax operator) in the inter-microphone processing modules. This sparsemax operator forces the model to mostly attend to one channel, hence learning implicitly to perform channel selection. Very recently, a work from Amazon Alexa [89] studies instead the similar problem of device arbitration. In this latter work the goal is to devise a system which can identify the best device that should respond to an user, provided that there are more than one smart device in the environment that received his query. To tackle this task a fully learnable end-to-end system is employed. This model is designed from the ground up to be efficient enough to run on on-edge devices.

3.2 MicRank: Channel Selection as a Learning to Rank Problem

In our work MicRank [66] we formulate the channel selection problem as a ranking problem. The goal is to devise a data-driven algorithm to perform channel selection e.g. for selecting the channel that minimizes the WER or, in general, a desired performance metric for the particular task at hand (it can also be a subjective measure such as mean opinion score (MOS)). Trivially, at first, one can think that this could be attained by simply training a classifier/regressor to predict e.g. the WER for each channel. However, we argue that this approach is sub-optimal: in fact, predicting the absolute value of the metric is not what matters in the end, we are only interested in the *relative performance* across the channels; and the training and inference strategy should be formulated to account for this important implication.

This can be achieved by training the model to rank the channels based on a pre-defined performance metric. An advantage of this approach is that this metric does not have to be differentiable (e.g. we can use the WER).

3.2.1 Learning to Rank

Learning to rank (LTR), is an established framework of information retrieval. The use of LTR-based algorithms is widespread: for example, they are behind most web search engines or social networks recommendation algorithms, such as the Twitter one[1].

The LTR field concerns with the development of algorithms that can automatically learn to rank items in a collection based on their *relevance* score to a specific user query. The goal is to obtain a ranking model that can accurately predict the relevance of a document, webpage, et cetera.

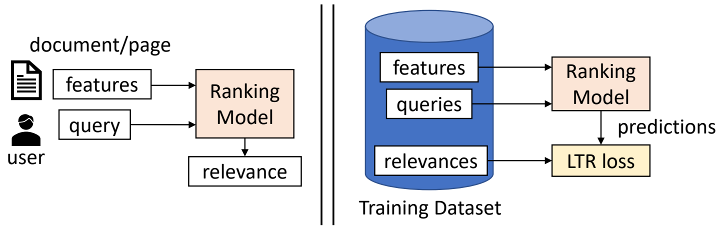


Figure 3.4: Learning to rank paradigm. Left: inference, which is performed on each item separately. Right: training, which instead is performed according to different strategies, usually taking into account 2 or more items at a time, as the model must learn to rank and not merely predicting relevances.

LTR typically uses supervised learning techniques to learn a ranking function from a set of training examples and the process is depicted in Figure 3.4. A training set is collected that consists in pairs of query-item instances, along with relevance labels and some features related to each item. Relevance scores indicate the degree of relevance of each item to the user query. During inference the model is applied to each query+item pair separately to retrieve the predicted relevances for each item, which can then be used to sort these latter. Instead, during training different strategies needs to be devised in order to train the model to learn to actually rank the items and not merely predicting the relevances of each item.

Throughout the years several training strategies and loss functions have been proposed to address this task. In our MicRank work we chose to explore two very popular strategies: RankNet and ListNet created specifically for DNN-based algorithms.

The LTR formulation for the purpose of ranking microphone channels has to be adjusted. Here, instead of an user query we will have a particular speaker utterance, our items will be the different observations at the different channels and, our relevance score will be some ASR-related performance metric, e.g. WER or word accuracy (WA). Note that in principle any performance metric could be used, and, crucially it can be non differentiable (as, in fact, WER or WA are). Here, WER or WA are used since we focus on ASR, but if the

[1] github.com/twitter/the-algorithm

back-end task is different, e.g. enhancement for an applications that required human listening, MOS could be used or any other enhancement-related metric.

3.2.2 The MicRank Framework

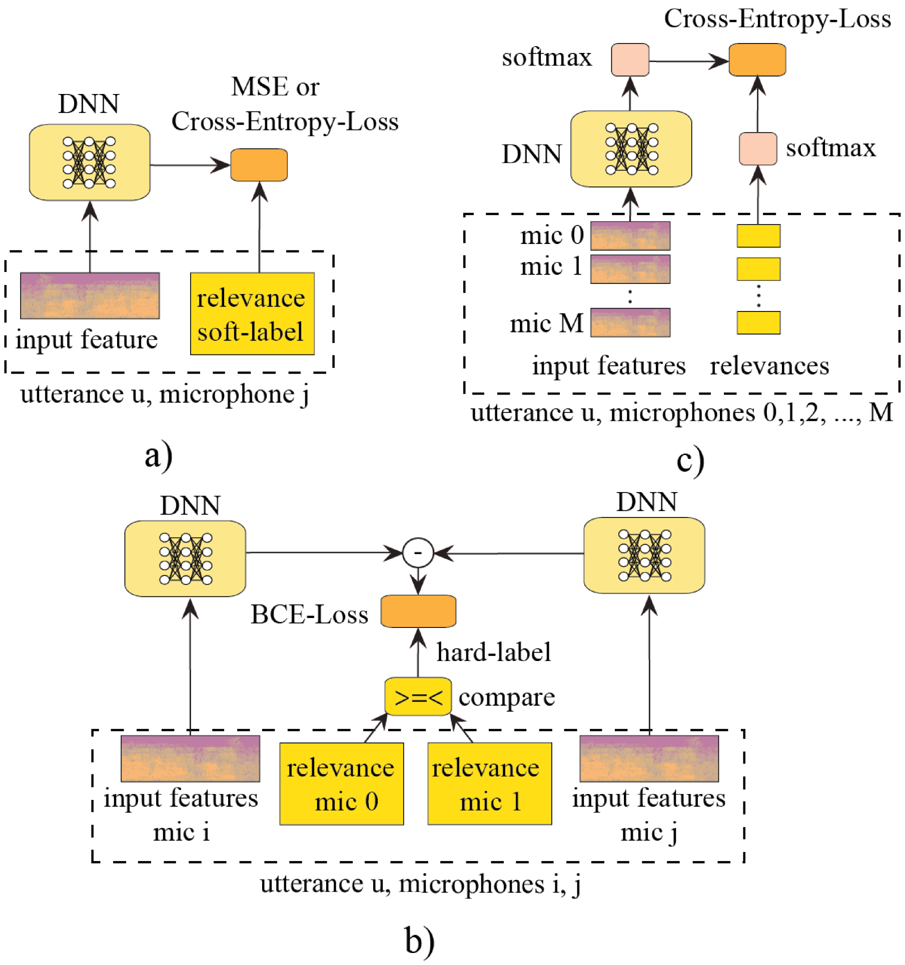


Figure 3.5: Training strategies: a) point-wise training; b) pair-wise training with RankNet; c) list-wise training with ListNet.

Let us assume that U utterances are recorded by M microphones. For each utterance u ($u = 0, \dots, U-1$), given the observation feature vector $\boldsymbol{x}_{u,i}$ for the i-th microphone ($i = 0, \dots, M-1$) and a ranking order (or relevance in information retrieval) $w_{u,i}$, our goal is to define a function $f(\boldsymbol{x}_{u,i})$ that generates the same ranking order: if $w_{u,i} > w_{u,j}$ then $f(\boldsymbol{x}_{u,i}) > f(\boldsymbol{x}_{u,j})$. In the following we describe different training strategies to achieve this goal, graphically depicted in Fig. 3.5.

Point-wise Training

The most straightforward approach to channel selection is to employ a model trained on each single channel individually to predict its relevance. In this method, given a set of training pairs $(\boldsymbol{x}_{u,i}, w_{u,i})$ for each utterance and microphone, the network is trained to minimize a cross-entropy loss:

$$\mathcal{L}_{\text{XCE}}^{point} = \sum_{u=0}^{U-1} \sum_{i=0}^{M-1} w_{u,i} \log\left[\sigma(f(\boldsymbol{x}_{u,i}))\right], \tag{3.1}$$

where $\sigma(\cdot)$ is the sigmoid operator. In this case, the relevance label $0 \leq w_j \leq 1$ is a soft label, representing the quality of the speech signal in an absolute term. WA for example, and any other bounded metric can be used straightforwardly. A clipping or normalization strategy instead can be adopted for metrics like WER which are unbounded or they can be modified accordingly to fit the $[0, 1]$ range. Alternatively, the cross-entropy training objective can be replaced by a Mean Squared Error (MSE) objective which does not require any bounded relevance assumption:

$$\mathcal{L}_{\text{MSE}}^{point} = \sum_{u=0}^{U-1} \sum_{i=0}^{M-1} \|w_{u,i} - f(\boldsymbol{x}_{u,i})\|^2. \tag{3.2}$$

Pair-wise Training

With point-wise training the model tries to learn to rank the channels by learning to predict their absolute quality. However, it does not consider relative performance of the other channels, as such it is a sub-optimal approach as it does not learn really to rank but only to predict their relevance. One way to account for the other microphones is to train the network in a pair-wise “siamese” fashion, as it has been proposed in RankNet [91]. In this case, labels are not required to represent an absolute measure, Thus even unbounded metrics can be used directly. For a given utterance u, let us consider feature vectors from two channels $x_{u,i}$ and $x_{u,j}$ with related relevance scores $w_{u,i}$ and $w_{u,j}$. We can define a binary pairwise label as:

$$y_{u,i,j} \begin{cases} 1 & \text{if } w_{u,i} > w_{u,j}, \\ 0 & \text{otherwise.} \end{cases} \tag{3.3}$$

Note that $y_{u,i,j}$ is an hard label (i.e. either 1 or 0) whose value depends on which relevance $w_{u,i}, w_{u,j}$ is higher than the other, and thus on the relative ranking of the two channels. For each training sample $(\boldsymbol{x}_{u,i}, \boldsymbol{x}_{u,j}, y_{u,i,j})$ we can then define a binary cross-entropy loss as:

$$\begin{aligned}\mathcal{L}_{u,i,j} &= y_{u,i,j}\log[P(w_{u,i} > w_{u,j})] \\ &\quad + (1 - y_{u,i,j})\log[1 - P(w_{u,i} > w_{u,j})],\end{aligned} \tag{3.4}$$

where $P(w_{u,i} > w_{u,j})$ is the probability estimated by the network $f(\cdot)$ that $\boldsymbol{x}_{u,i}$ is more relevant than $\boldsymbol{x}_{u,j}$, which can be computed as:

$$P(w_{u,i} > w_{u,j}) = \sigma\left(f(\boldsymbol{x}_{u,i}) - f(\boldsymbol{x}_{u,j})\right). \tag{3.5}$$

The overall training loss is obtained by summing over all unique microphone pairs and utterances:

$$\mathcal{L}^{pair}_{\mathrm{BCE}} = \sum_{u=0}^{U-1} \sum_{(i,j)\in\mathcal{I}_u} \mathcal{L}_{u,i,j}. \tag{3.6}$$

where $\mathcal{I}_u = \{(i,j) : |w_{u,i} - w_{u,j}| > \delta, i \neq j\}$ is the set of microphone pairs whose relevance difference in utterance u is larger than δ with $\delta \geq 0$. Note that in Eq. 3.6 we consider only pairs where one of the channels is more relevant than the other and discard Note that all pairs where channels have the same or very similar relevance (in our case WA or WER) are discarded. Thus the size of the training set is upper bounded to $(U-1)(M-1)(M-2)/2$.

List-wise Training

In RankNet, the network learns to order the items by comparing them in a pairwise fashion during training. However, due to the use of hard labels, the learning process does not take into account the actual difference between two samples as it cares only for relative pair-wise ordering. But, intuitively, swapping the ranks of two samples with very similar relevance should be less critical than swapping two samples with a very different relevance.

These problems can be addressed by employing ListNet [92]. Contrary to the pair-wise approach, for each utterance u all available microphones M are used to compute a cross-entropy loss:

$$\mathcal{L}^{list}_{\mathrm{XCE}} = \sum_{u=0}^{U-1} \sum_{i=0}^{M-1} \mathcal{S}(w_{u,i}) \log[\mathcal{S}(f(\boldsymbol{x}_{u,i}))]. \tag{3.7}$$

$\mathcal{S}(\cdot)$ is the softmax operator which ensures that both labels and network outputs can be treated as probability distributions. It also enforces that ranking, for each utterance, is determined only by relative performance of each microphone. The total number of examples in ListNet is simply the number of utterances in the training set $U-1$ as channels are considered all together in the loss.

3.3 Datasets

In this Chapter we used three different datasets in our experiments: a synthetic dataset generated on purpose, the CHiME-6 Challenge dataset and the very recent CHiME-7 DASR Challenge dataset. We describe them in detail thereafter.

3.3.1 Synthetic Dataset

We generated a multi-channel synthetic dataset featuring an ad-hoc network with 8 cardioid microphones randomly scattered inside a room. Clean speech utterances are uniformly sampled from LibriSpeech [7] using `train-clean-100` for training, `dev-clean` for validation and `test-clean` for test. We used a total of 20k utterances for train and 2k for validation and test splits. Point-source noise from the dataset in [93] is also employed to make the data more realistic. A different acoustic scenario is sampled for each utterance. Using gpuRIR [94] we simulate a rectangular room whose size and reverberation time (T60) are sampled uniformly between 10 and 60 m^2 and between 0.2 and 0.6 s respectively. The positions and orientations of the speaker, noise and of the 8 microphones are chosen randomly inside the room but with the constraints that the speaker cannot be closer than 0.5 m from any microphone or wall and each microphone should be at least 0.5 m apart from any other.

Relevance labels are obtained by training an ASR on the training set using an opportunely modified Kaldi [95] LibriSpeech recipe[2], and then decoding and computing the errors (insertions, deletions etc.) on such set.

3.3.2 CHiME-6

The CHiME-6 dataset features real dinner parties. The recordings are divided into 20 sessions for a total of more than 60 h of data. In each session, 4 speakers are recorded in a real home environment usually across different rooms. Due to the particular setting, it features conversational speech and low Signal-to-Noise Ratio (SNR). Recordings from binaural microphones worn by each speaker are provided along with distant speech captured by 6 array devices with 4 microphones each for a total of 24 microphones. Two different annotations are provided for the start and end time of every utterance: looser ones geared towards Automatic Speech Recognition (ASR) and tighter ones obtained via forced-alignment. The latter ones are more suitable for evaluating VAD and diarization systems and will be used in Chapter 5. Here instead, as we focus on ASR, we use the former.

[2] `https://github.com/kaldi-asr/kaldi/tree/master/egs/librispeech/s5`

To perform our channel selection experiments we employed the ASR backend provided by the challenge organizers as the baseline system. We used the official Kaldi recipe[3] with the acoustic model and the two-pass decoding in [96].

3.3.3 CHiME-7 DASR

The CHiME-7 DASR challenge dataset is comprised of three different scenarios/sub-datasets: CHiME-6, DiPCo [97] and Mixer 6 Speech [98]. CHiME-6 was already described in the previous section, and here is used unaltered with the exception that two sessions were moved from training to evaluation and the text normalization was changed a bit to standardize non-words expressions such as “mmh”, “mhm” etc. DiPCo features a similar scenario as CHiME-6, a dinner party among 4 participants but in DiPCo everything takes place in the same room (which does not change between development and evaluation). It comprises of 10 sessions recorded by 5 far-field devices each with a 7-mic circular array. The 10 sessions are divided equally between development and evaluation. No training partition is provided. Mixer 6 Speech instead setting is different from the aforementioned two as it consists in 2-persons interviews recorded in a room with multiple heterogeneous recording devices.

The goal of this challenge is to have participants devise meeting transcriptions systems that can work across multiple scenarios featuring different arrays, varying number of participants and diverse acoustical conditions.

3.4 Experimental Analysis

3.4.1 Synthetic Dataset

Results on the synthetic dataset are reported in Table 3.1. The first row in the Table reports results obtained by randomly selecting one of the microphones. Following we report also the results obtained using different oracle measures. Interestingly, we can see that picking, each time, the closest microphone to the source, while of course better than random choice, it is not the best strategy among all the oracle strategies. In fact, STOI computed with respect to the oracle anechoic speech signal provides the best result. However, we can also see that all signal-based measures and closest are still very far from the true oracle (selection made based on WER).

Among the baseline blind channel selection techniques, EV and AM-Entropy considerably improve over random selection. They are generally quite effective as the performance drop is very modest compared to the best signal-based oracle measures.

[3] `https://github.com/kaldi-asr/kaldi/tree/master/egs/chime6/s5_track1/local`

Table 3.1: WER on the synthetic dataset. We report both the best WER as well as the average WER on the Top-3 selected microphones.

Ranking Method		**Dev**		**Test**	
		Best	Top-3	Best	Top-3
	Random Selection	51.7	51.5	40.9	41.1
oracle	CD-Informed [82]	45.1	47.7	36.9	38.3
	PESQ	41.9	45.8	33.1	36.4
	closest	37.0	45.1	29.9	36.1
	SDR	37.4	43.8	29.6	34.9
	STOI	36.3	44.2	29.2	35.2
	WER	32.0	39.6	24.8	30.6
baseline	CD-blind [82]	46.1	48.1	36.2	39.4
	EV [81]	39.0	44.9	31.8	35.8
	AM-Entropy [87]	41.2	45.8	31.1	35.5
MicRank	Point-wise XCE	37.3	44.1	30.4	34.6
	Point-wise MSE	36.9	43.7	30.0	34.3
	RankNet	36.5	43.4	28.8	34.1
	ListNet	**36.0**	**43.2**	**28.5**	**33.9**

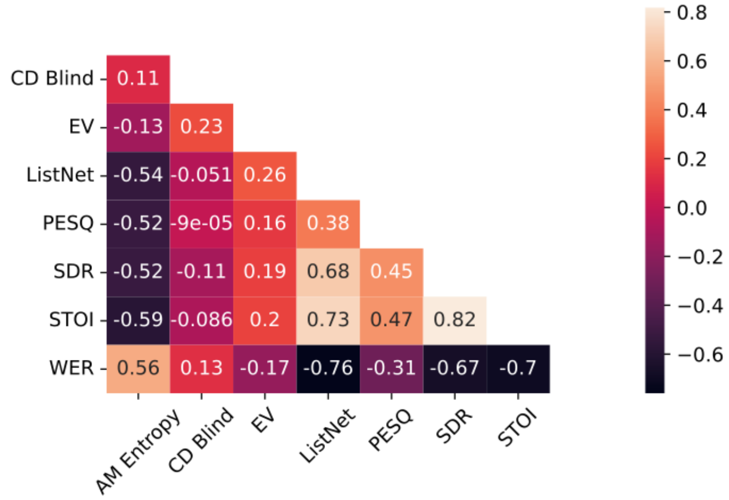


Figure 3.6: Pearson correlation plot between the different channel selection techniques on synthetic data.

All MicRank-based techniques are able to bring substantial gains over such previous blind selection methods. In particular, we can observe that, as expected, pair-wise and list-wise methods outperform point-wise ones which cannot account for relative performance. Notably, the best WER for RankNet and ListNet is lower than the Top-3 averaged WER of oracle WER selection, indicating that these methods are able to pick up always the best or second-best channel among the top 3. Amidst previously proposed selection methods, EV and AM-Entropy have comparable performance despite the former is remarkably less computational expensive.

In Figure 3.6 we report a Pearson correlation plot for a subset of selection metrics obtained on synthetic dataset test set. Interestingly, EV has rather low correlation with WER despite properly selecting favorable channels as shown in Table 3.1. We observed that EV fails to rank the channels with high WER. CD-Blind has the same behaviour while AM-Entropy, which is posterior based, shows much better correlation even for unfavourable channels. Again, we can notice that the proposed method is the one with highest absolute correlation value and surpasses even some oracle measures.

Table 3.2: WER on CHiME-6 development and evaluation sets.

Ranking Method		**Dev**	**Eval**
		Best	Best
	Random Selection	73.1	68.0
oracle	CD-Informed [82]	70.8	68.7
	PESQ	66.0	60.1
	SDR	65.2	58.9
	STOI	64.8	58.5
	WER	56.7	51.3
	CHiME-6 Baseline	69.2	60.5
baseline	CD-blind [82]	72.5	67.0
	EV [81]	68.6	59.9
MicRank	RankNet	67.4	**59.0**
	ListNet	**67.2**	59.5

3.4.2 CHiME-6

Finally, in Table 3.2 we report the performance achieved on CHiME-6 data for the most promising approaches as found on the synthetic set. We can see that both EV and MicRank methods considerably improve with respect to the CHiME-6 Baseline, which benefits from “pseudo-oracle" knowledge of the speaker position and features dereverberation plus beamforming.

Both RankNet and ListNet based systems improve over EV but, contrary to the synthetic dataset, are unable to outperform signal-based oracle-level performance especially on the development set. This is mainly due to the fact that CHiME-6 features a substantial amount of overlapped speech [99], while in the synthetic data only one speaker is present. And, in fact overlapped speech is particularly high in the development set, which is where we observe the largest difference between signal-based oracles and the proposed method. Current selection methods, including MicRank, are unable to account for speaker identity when ranking the channels for a given utterance. This can lead to mistakenly rank the channels with respect to the interfering speaker instead of the desired one, leading to considerable degradation in ASR performance.

On the other hand, signal-based oracle measures are able to implicitly account for this because they are computed with respect to the correct speaker close-talk microphone. RankNet seems to generalize better than ListNet on

CHiME-6 due to the fact that on CHiME-6 relevances are very close to each other in the training set but not in the dev and eval sets. In this scenario, using hard labels, as in RankNet, could help boosting discriminability and generalization, and may help during training as the gradient is stronger.

3.5 Further Analysis on Channel Selection

We present in the following one further study on this matter, which show promising research directions.

3.5.1 CHiME-7 DASR Baseline: Combining EV Ranking with Guided Source Separation

As an additional use case for channel selection, we report here a study regarding the use of EV selection in the contest of the recent CHiME-7 DASR Challenge of which we are co-organizers.

In particular EV is used in the challenge baseline system for the acoustic robustness sub-track, which allows the use of oracle diarization and whose ranking score is based on speaker-attributed WER (SA-WER). In the development of the baseline system we focused on the use of EV for improving the performance and inference time of the already effective guided source separation (GSS) algorithm, which performs multi-channel semi-blind target speaker enhancement prior to ASR transcription. Such baseline model is composed of three components and it is summarized in Figure 3.7: EV selection, GSS and ASR. EV is used to select a promising subset of microphones which are then used for GSS. The output of GSS is then used for recognition.

The main reason for using EV, instead of MicRank, was due to the fact that the baseline system had to be simple and it was also motivated by the results in the previous experiments on CHiME-6, where it achieved results very close to MicRank.

In Table 3.3 we report the results of such system in terms of speaker-attributed WER (SA-WER), for both development and evaluation sets. Results are reported separately for each of the three CHiME-7 DASR scenarios: CHiME-6, DiPCo and Mixer 6. Note that, unfortunately, the machine on which the experiments were run was shared with other users and we didn't perform multiple runs to account for random factors of variations (e.g. computational load due to other users), thus the time figures are indicative.

Nonetheless, we can clearly see some trends, which are made more evident by plotting in Figure 3.8 and in Figure 3.9 the values in the Table 3.3 above, respectively for the development and evaluation sets.

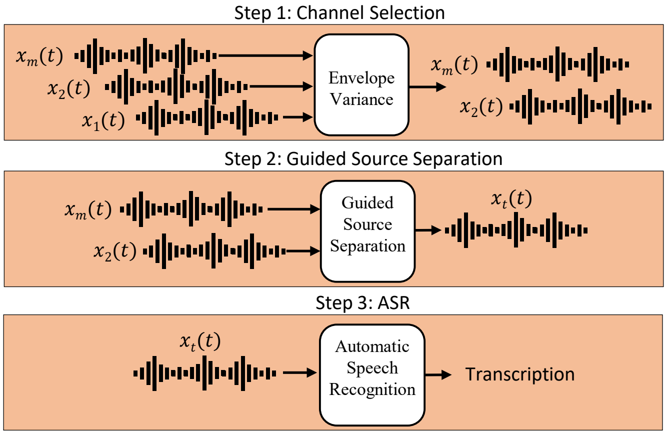


Figure 3.7: Block scheme for the CHiME-7 DASR acoustic robustness sub-track baseline. GSS uses oracle diarization in this case.

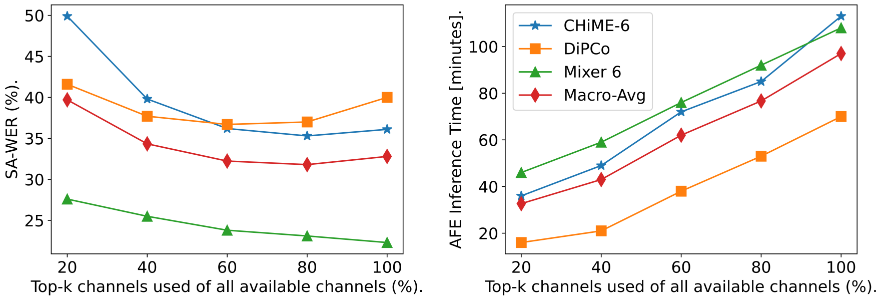


Figure 3.8: Results on CHiME-7 DASR development set for each scenario: CHiME-6, DiPCo and Mixer 6, plus macro-average across all three. Left: SA-WER, right: GSS+selection inference time. X-axis: percentage of the ranked channels used.

Table 3.3: CHiME-7 DASR results for each scenario: SA-WER and GSS+selection inference time versus EV top-k microphones used; (%) of all available channels.

Top-k (%)	**Scenario**	**Dev**		**Eval**	
		SA-WER (%)	Time (h:mm)	SA-WER (%)	Time (h:mm)
100	CHiME-6	36.1	1:53	38.9	3:30
	DiPCo	40.0	1:10	41.5	1:10
	Mixer 6	22.3	1:48	27.0	0:52
80	CHiME-6	35.3	1:25	38.4	2:58
	DiPCo	37.0	0:53	40.3	0:46
	Mixer 6	23.1	1:32	29.5	0:31
60	CHiME-6	36.2	1:12	38.6	2:12
	DiPCo	36.7	0:38	40.3	0:29
	Mixer 6	23.8	1:16	33.8	0:19
40	CHiME-6	39.8	0:49	40.2	1:30
	DiPCo	37.7	0:21	42.0	0:19
	Mixer 6	25.5	0:59	36.0	0:18
20	CHiME-6	49.9	0:36	48.7	1:05
	DiPCo	41.6	0:16	47.1	0:15
	Mixer 6	27.6	0:46	37.4	0:14

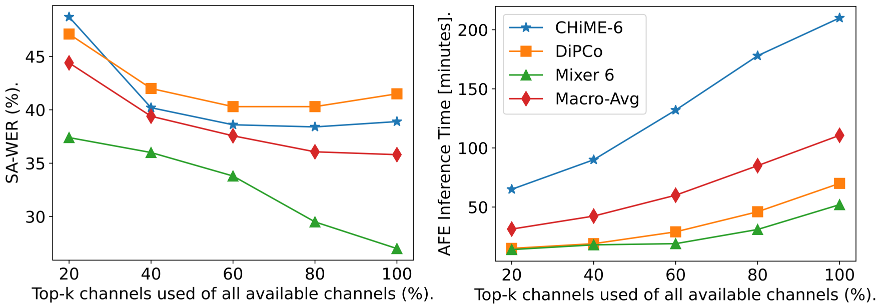


Figure 3.9: Results on CHiME-7 DASR evaluation set for each scenario: CHiME-6, DiPCo and Mixer 6, plus macro-average across all three. Left: SA-WER, right: GSS+selection inference time. X-axis: percentage of the ranked channels used.

First, the inference time seems to grow approximately in a linear manner with respect to the number of channels used. At the same time however, generally the SA-WER decreases as the number of channels increases. There is thus a trade-off between computational complexity and performance. Only on the two scenarios of CHiME-6 and DiPCo it appears that selecting a subset also increases performance: the best results are achieved by the top-k 80% configuration. These two are also the two nosiest datasets and with most speech overlap, thus it makes sense that in this scenario excluding the most problematic channels could bring an improvement. On the contrary Mixer 6 is much less noisy and all channels can contribute significantly in improving the GSS result.

This study reinforces our remarks, made in Section 3.1 about the fact that channel selection/ranking can be used also in conjunction with other AFE methods to lower the computational requirements, but also, in some instances improve performance. And, in fact, one of the main goals of the proposed CHiME-7 DASR challenge is to try to devise new, better channel selection and ranking algorithms as the participants are forced to devise a single system that is able to generalize to all three scenarios.

3.6 Conclusions & Future Work

In this Chapter, we presented an overview of the channel selection problem from the first techniques based on signal, decoder and acoustic model measures to the latest ones, which instead rely on data-driven, machine learning approaches.

We then made a compelling argument for framing the channel selection as a Learning to Rank (LTR) problem and we outlined our work, MicRank, in which we address channel selection with LTR techniques. Such work is also the first proposing a data-driven, deep-learning based method to address this problem. We explored three different LTR training strategies: point-wise, RankNet and ListNet and validated our method on a synthetic dataset and CHiME-6.

We showed that the proposed method is able to outperform previous state-of- the-art channel selection approaches and, on single-speaker synthetic data, is even able to surpass oracle signal-based selection. It is very lightweight compared to SSE methods, agnostic to the array topology and robust to misalignment across the microphone channels as it can run independently on each device. Its modest computational requirements could allow to run it on each device separately and save bandwidth as only the relevance scores needs to be uploaded server side and then compared. The best channel could then be uploaded to the ASR pipeline in the cloud.

Results on multi-speaker real-world CHiME-6 data suggest that there are still some challenges to overcome. In fact, we found very marginal improvement over effective signal-based methods such as EV, as there is inherently ambiguity in the channel selection problem when two speakers overlap. For such reason it may be worth exploring conditioning via target speaker-id embeddings as this could help the method to learn to disambiguate between the target and the interferer speakers. Another future direction could be assessing generalization across different ASR models and scenarios, as well as a stremable extension.

As a further study we also reported an ablation study over the CHiME-7 DASR acoustic robustness sub-track baseline, on the use of EV for the purpose of channel subset selection prior to GSS multi-channel target-speaker enhancement. The results here indicate that channel selection could bring also in this instance benefits in terms of performance and reduced computational requirement.

Chapter 4

Robust Keyword-Spotting via Implicit Acoustic Echo Cancellation

Context

This work was presented at SLT 2022 [100] and was made as an applied scientist intern at Amazon Alexa with Thomas Balestri and Thibaud Sénéchal from the Wakeword team. It was done in Cambridge, USA, during summer 2021 and I had a great time there.

4.1 The “Barge-in” Problem

In many speech-enabled human-machine interactions, the user speech can overlap with the device playback audio [101–103], in the same way as e.g. it happens between human-to-human interactions, where one interlocutor begins speaking before another has finished. This phenomenon, illustrated in Figure 4.1 and colloquially called “barge-in” [101, 103], is very challenging since the signal-to-interferer ratio (SIR) between the user speech and device playback is usually very low due to the fact that the device loudspeakers are closer to the microphones than the user is.

Moreover, when the playback device audio consists in a TTS generated response or podcast audio many tasks can become ill-defined/ambiguous and thus performance could drop. For example, considering custom/multi-KWS [104, 105], without appropriate countermeasures, a model will be prone to pick up also keywords from the TTS playback, especially as TTS models are increasingly realistic or include celebrity-derived custom voices. This can lead to the device “self waking" and continuously interrupting itself as the model, alone, cannot implicitly distinguish between user and device speech and ignore this latter. Such problem also affects ASR or keyword-less initiated interactions, such as DDD[106–109], and is actually even more serious in these cases due to the fact that both are open vocabulary. One trivial way to mitigate this issue would be disabling the KWS functionality while the device is in playback. Yet, doing so prevents the user to “barge in", making the interaction significantly less natural and intuitive for the user.

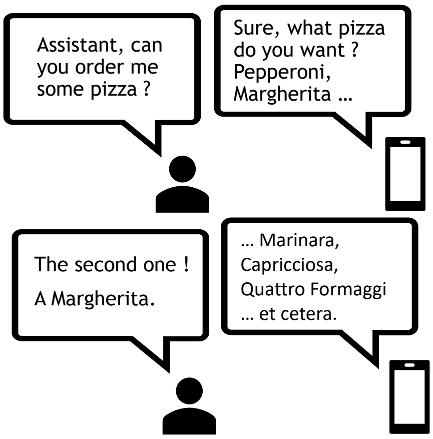

Assistant, can you order me some pizza ?
Sure, what pizza do you want ? Pepperoni, Margherita ...
The second one ! A Margherita.
... Marinara, Capricciosa, Quattro Formaggi ... et cetera.

Figure 4.1: A picture is worth one thousand words: the “barge-in” problem is when the user tries to talk over the device playback, being it TTS, music or anything else.

Following [101–103], this problem is best formulated in the acoustic echo cancellation (AEC) framework as usually the device playback audio is known. We can thus define the playback *reference signal* $\boldsymbol{r}$ and a mixed signal $\boldsymbol{y} = \Gamma(\boldsymbol{r})+\boldsymbol{u}$, where $\Gamma(\cdot)$ is a (possibly non-linear) function and $\boldsymbol{u}$ is the *target signal* for the task at hand i.e. the signal which would be captured by the device if there wasn’t any playback return signal $\Gamma(\boldsymbol{r})$. This can include instances where there is no user “barging in", i.e. for which $\boldsymbol{u}$ is only background noise, which the classifier should ignore.

A common model for $\Gamma(\cdot)$ is to use a linear approximation such that $\boldsymbol{y} = \boldsymbol{r} * \boldsymbol{h} + \boldsymbol{u}$, where $\boldsymbol{h}$ is the impulse response that characterizes the propagation of $\boldsymbol{r}$ and includes effects from the room, speaker, and microphone. We will refer to $\boldsymbol{n} = \Gamma(\boldsymbol{r})$ as the *interferer signal* hereafter. $\boldsymbol{u}$ itself could be the user far-field speech with reverberation and other interferers.

AEC techniques can be employed to obtain an estimate of the target $\boldsymbol{u}$ by leveraging the reference signal $\boldsymbol{r}$. These include both classical [101–103, 110–114] and neural-based (nAEC) methods [105, 115–119] which are generally more effective. These latter however require the oracle target signal to be available at training time, which is difficult to obtain directly on real-world data, at least in a scalable way. Thus, nAEC methods rely on synthetic data for training, which is inherently mismatched with respect to real-word data. To counter this mismatch, [105] proposes the use of an additional ASR auxiliary loss obtained from the latent representation of the encoder. Instead [104] leverages ASR in inference and proposes a framework for cancelling the TTS playback interferer $\boldsymbol{n}$ by using the textual source of the reference TTS signal. However, this latter is only applicable to TTS playback and does not generalize to other forms of playback audio such as podcasts or music. In addition, nAEC methods that rely on ASR cannot be used for always-on frontend applications such as KWS since it would be too computationally expensive to perform ASR continuously on resource-constrained edge devices.

A computationally-effective way to address this problem, suitable for KWS and DDD tasks on edge devices, could be devised if we directly feed the reference signal $\boldsymbol{r}$ as an additional input to the back-end task model together with the mixture signal captured by the device. The main idea is to give the KWS or DDD classifier access to the reference signal to allow to disambiguate between the target and the playback return signal and learn to ignore this latter without the need for an AEC or nAEC pre-processing front-end. If the fusion strategy is designed well, as we will see, it could allow for considerable computational resources savings as the computational overhead for having a classifier taking also the reference signal in input is extremely low.

Such approach was explored with the Amazon Alexa Wakeword team and presented at SLT 2022 [100]. We called it *implicit Acoustic Echo Cancellation* (iAEC) and it led to very promising results. In particular, we studied two different strategies for feeding the reference channel to the back-end classifier, one that involves concatenation and another based on latent-representation masking. We found this latter choice especially promising as it allows for the KWS and DDD architecture to be unchanged (no computational overhead) when there is no playback, which is the predominant scenario in deployment.

Moreover, our work explored DDD for always-on, streaming scenarios for the first time. Previous DDD works [106–109] performed DDD downstream of a KWS model. These systems are not always running, are more resource intensive, and have higher latency than front-end components. Here instead we consider the use of a DDD model that is run continuously and subsumes the role of a KWS model, allowing for a full keyword-free interaction. Understandably, as it is keyword-free and continuously running, such model is especially affected by the device playback issue and prone to the “self wake" issue and thus addressing this task is a necessity for enabling such new DDD application.

4.2 Implicit Acoustic Echo Cancellation

As said, the focus of our iAEC work is to address the KWS and DDD tasks with the goal of devising a computationally efficient strategy to improve the performance during device playback without incurring in a degradation in normal conditions (non-playback).

During playback, a KWS or DDD classifier generally observes only the mixture $\boldsymbol{y} = \Gamma(\boldsymbol{r}) + \boldsymbol{u}$. As said, this presents a challenge when the interferer $\boldsymbol{n} = \Gamma(\boldsymbol{r})$ and target $\boldsymbol{u}$ could lead to an ambiguity for the task at hand (e.g. the interferer $\boldsymbol{n}$ contains a keyword but the target $\boldsymbol{u}$ does not). In such situations, training a classifier on the mixture signal $\boldsymbol{y}$ without access to the reference could bias the model to learn unintended characteristics of the interferer signal. For example, the TTS voice/gender or the fact that the playback interferer has usually more energy than the user speech as the loudspeakers are close to the microphones. These biases could degrade performance severely when the model is deployed. In addition, if the interferer and target signals come from the same distribution (e.g. gender-unbiased overlapping human voices), the task becomes impossible since the model cannot differentiate between the target and interferer signals. One way to obviate to this is to feed to the KWS or DDD classifier the reference signal along with the mixture signal ones to aid in the classification task. The name iAEC stems from the fact that the model here does not have to explicitly recover the target signal as in AEC methods (an arguably more complex task), but only learn how to use the additional reference input $\boldsymbol{r}$ to ignore the playback return signal $\boldsymbol{n}$ and classify correctly the target signal $\boldsymbol{u}$.

In this framework, the goal is to learn a function $\mathcal{F}(\boldsymbol{y}, \boldsymbol{r}, \boldsymbol{\theta})$ parametrized by $\boldsymbol{\theta}$ that models the joint conditional distribution $P(y_\mathcal{T}|\mathbf{y}, \mathbf{r})$ where $y_\mathcal{T}$ are the labels for the task at hand belonging to the target signal $\boldsymbol{u}$ (e.g. for KWS, keyword or non-keyword). Importantly, this formulation includes non-playback conditions, e.g. for which the reference $\boldsymbol{r}$ and thus the interferer $\boldsymbol{n} = \Gamma(\boldsymbol{r})$ is zero and the mixture signal is simply equal to the target $\boldsymbol{y} = \boldsymbol{u}$. For such instances the problem simply resolves to modeling $P(y_\mathcal{T}|\mathbf{u})$ as in “classical" KWS or DDD.

4.2.1 Reference Signal Fusion

Concatenation: iAEC-C

Since we focus in this work on KWS and DDD tasks, where usually features like log Mel-filterbank energies (LFBEs) are employed, we can concatenate the mixture and reference signals feature vectors, respectively $\mathbf{x}_y(k)$ and $\mathbf{x}_r(k)$, and feed them to the classifier. To make notation less cumbersome, we drop the frame index k in the following. This simple method could allow the classifier to exploit the reference channel information.

As depicted in Figure 4.2, left panel, during playback mode, we apply batch normalization [120] (BN) separately to the reference and mixture branches prior to concatenation. This is done because reference and mixture signals

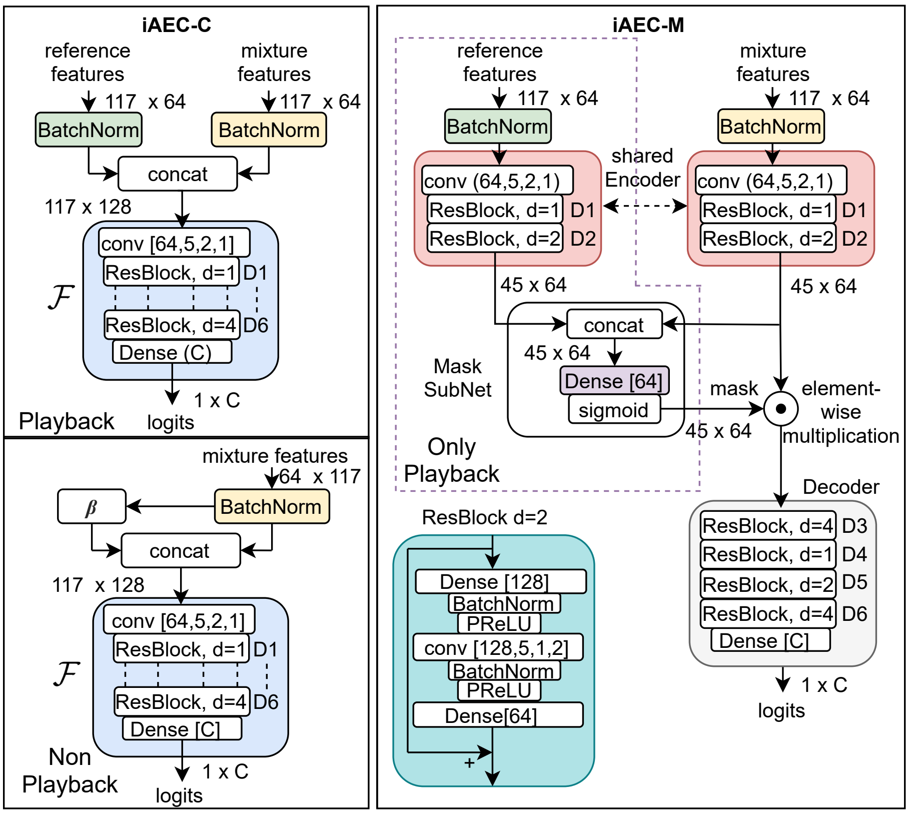


Figure 4.2: Schematic of implicit acoustic echo cancellation with concatenation (iAEC-C) and encoder-masking-decoder (iAEC-M) architectures. The LFBE feature extraction part is omitted for simplicity. Neural blocks and input features are described in detail in Section 4.4.1. Tensor dimensions are represented as *sequenceLength* × *featureMaps* and we display the values used during training. 1D convolutional blocks (conv) are represented as [*featureMaps*, *kernelSize*, *stride*, *dilation*] while linear layers (Dense) as [*featureMaps*]. Regarding iAEC-M, we show the best configuration (D2) as found in Section 4.4.2. We also report for convenience the layers used in each ResBlock, which has the same structure as in [53, 99].

have usually very different gains, as the latter is often far-field speech. During non-playback conditions, the input is concatenated with the learned bias parameters β of the BN layer. This is equal to concatenating the mean of the post-normalized input features (see left-bottom panel of Figure 4.2).

Masking: iAEC-M

Concatenating the input feature with the learned β parameters from the BN layer is a waste of computing resources in non-playback conditions, which account for most of the deployment time for always on, on-device applications. This problem can be obviated by using the reference to produce a sigmoid mask which is applied to the latent representation of the mixture signal, as illustrated in the right panel of Figure 4.2.

Compared to iAEC-C, here $\mathcal{F}$ is split into an encoder $\mathcal{E}$ and decoder $\mathcal{D}$. The encoder is shared between reference and mixture branches to save L1 cache memory and produces two latent representations: the reference embedding $\boldsymbol{Z}_r = \mathcal{E}(\boldsymbol{x}_r)$ and the mixture embedding $\boldsymbol{Z}_y = \mathcal{E}(\boldsymbol{x}_y)$.

These latent representations are $\in \mathbb{R}^{T \times D}$, where T is the sequence length and D the embedding size. They are concatenated and used to derive a mask through a linear projection layer $\mathcal{P}$ with weight matrix $\in \mathbb{R}^{2D \times D}$, followed by a sigmoid activation $\sigma(\cdot)$: $\boldsymbol{M} = \sigma(\mathcal{P}([\boldsymbol{Z}_y, \boldsymbol{Z}_r]))$. This mask is then applied to the encoded representation from the mixture branch $\boldsymbol{Z}_\gamma = \boldsymbol{M} \odot \boldsymbol{Z}_y$ via element-wise multiplication. This operation acts as a gating mechanism over $\boldsymbol{Z}_y$. Finally $\boldsymbol{Z}_\gamma$ is fed to $\mathcal{D}$ to obtain the predictions. In non-playback mode the masking mechanism along with the entire reference branch are dropped and $\boldsymbol{Z}_\gamma = \boldsymbol{Z}_y$ directly.

4.2.2 On-the-fly Data Augmentation

In some application scenarios the reference playback signal $\boldsymbol{r}$ from the device may be not available for training and only the mixture $\boldsymbol{y}$ is available. In edge-applications for example, the playback TTS signal is usually not uploaded to the server-side to save bandwidth. Moreover, regarding smart-home devices, indeed most of the examples available in training can feature limited or no user barge in scenarios as e.g. could be an experimental/not fully supported feature. In these instances one obvious solution is to add artificial device playback via simulation. This may require a complex pipeline involving room, loudspeaker and front-end simulations and still lead to sub-optimal results as the simulated data will be mismatched with respect to the real-world one.

Instead, here, as an additional contribution, we propose a simple but effective on-the-fly data-augmentation strategy to generate $(\boldsymbol{y}, \boldsymbol{r}, \boldsymbol{n})$ triplets by sampling multiple mixture signals $\boldsymbol{y}$ from a training set which can also contain only legit

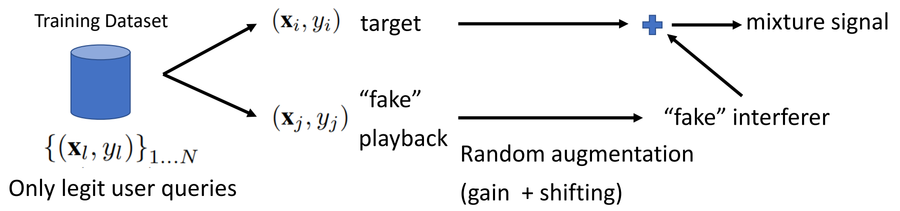


Figure 4.3: Proposed on-the-fly data augmentation strategy.

user queries example. It is depicted in Figure 4.3 and explained in detail in the following.

Given a collection of N training examples $\mathbf{x}_l$ and corresponding labels y_l for the task at hand $\{(\mathbf{x}_l, y_l)\}_{1\ldots N}$ we randomly sample two examples $(\mathbf{x}_i, y_i)$ and $(\mathbf{x}_j, y_j)$. These two examples are arbitrarily assigned the role of target $\boldsymbol{u} = \mathbf{x}_i$ and reference $\boldsymbol{r} = \mathbf{x}_j$ no matter their original labels. We then generate the mixture $\boldsymbol{y}$ by applying a random time-shift to $\mathbf{x}_j$ and mixing it with $\mathbf{x}_i$ at a randomly chosen SIR. The corresponding mixture $\boldsymbol{y}$ is assigned label $y_{\mathcal{T}} = y_i$ and the interferer label y_j is ignored. This strategy forces the model to learn to ignore $\mathbf{x}_j$ by using the reference $\boldsymbol{r}$, whatever the original label y_j. As $\mathbf{x}_j$ can contain a legit keyword from a user, without the reference $\boldsymbol{r}$ it would be impossible for the model to ignore the interferer signal (as it would be a legit user keyword) and output the correct prediction relative to the target $\boldsymbol{u}$. Because the target and interferer belong effectively to the same distribution, this data augmentation can also mitigate potential bias in the training data and boosts generalization to new speakers as we show in Section 4.4.2. For example, if artificial reverberation is used in the simulation of the interferer signal, this could introduce some bias that the model can explore. Here, instead we only apply gain augmentation and shifting minimizing such possible sources of bias.

In practice it is also possible to leverage unlabeled data when assigning the interferer, as for this role the label is not needed. However we did not explore such possibility in our experiments here, demanding it to future works.

We show that, as far as on-device KWS and DDD applications is concerned, this naive technique is competitive with simulation via image-method techniques such as gpuRIR [94] and, used as an additional data-augmentation strategy, can improve the results even when the true oracle target, interferer and reference are available in training (see Section 4.4.3). We show in Section 4.4.3 that it can also boost performance of nAEC models and, in Section 4.4.2 that can help reduce potential biases in the dataset related to the TTS models used and their perceived gender.

4.3 Datasets

We used two datasets for our experiments. One, fully synthetic is derived from the popular KWS benchmark dataset Google Speech Commands v2 (GSCv2). The second one, instead, is courtesy of Amazon Alexa and was used when the Author was an intern there with the wakeword team. We describe both thereafter.

4.3.1 Speech Commands Mix

To study how playback can degrade KWS in a controlled scenario, we extended GSCv2 [121] with TTS and music. We generated two additional versions of the original dataset by mixing TTS from LibriTTS [122] and music from Musan [123], respectively. To generate challenging TTS interferers, we sampled segments from LibriTTS containing GSCv2 keywords. The temporal location of the keywords from LibriTTS was determined using forced alignment. Interferer and reference pairs are generated using gpuRIR [94] by adding artificial reverberation to the original LibriTTS segments. For each room impulse response, we sampled a room size from uniform distribution $U(10, 50)$ m^2 and T60 reverberation time from $U(0.2, 0.6)$ s. The position of the source is chosen randomly inside the virtual room. To simulate a smart-speaker device, the microphone position is constrained to be in a radius of 5 cm from the source, oriented outward with cardioid polar pattern. Mixture files are obtained by mixing the reverberated interferer signal and original GSCv2 signal with SIR $\sim U(-12, 3)$ dB, which is consistent with what is observed on Alexa devices. An equivalent procedure is followed for mixtures with music. We use the official GSCv2 training, development and test split in our experiments. The scripts to generate this dataset were made available[1].

4.3.2 Alexa “Follow-Up Mode" Dataset

Amazon Alexa assistant provides a “follow-up" mode (FUM) [108] that allows users to interact with the device agent without repeated use of the wakeword. We leverage this data corpus for training and evaluating our DDD model. Ground truth DDD speech annotations are provided for each FUM utterance. The dataset consists of 538k utterances split into train, dev, and test partitions of 465k, 20k, and 53k utterances, respectively. As FUM does not contain any playback data, in our experiments we also used a TTS model to generate synthetic playback signals for the default Alexa voice. In our experiments we also report results when another, unmatched voice (*Matthew*) is used in training. We generated 500k clean TTS utterances and used gpuRIR to add artificial

[1] `https://github.com/popcornell/SpeechCommandsMix`

reverberation with the same configuration described in Section 4.3.1. For evaluating our models under TTS playback conditions, we used an Alexa Echo Show 10 device to record both playback-only device responses and instances where 10 speakers were tasked to say commands over the TTS audio. The default TTS Alexa voice was used. This lab collection resulted in a corpus of approximately 2 hours and 45 minutes which was split equally in development and test partitions. Note that both training/dev and test sets can contain background environmental noise in the target signal.

4.4 Experimental Analysis

4.4.1 Architecture and Training Details

For our experiments we employ the Temporal Convolutional Network (TCN) from [99] with a few modifications. Firstly, we use here a smaller network with $X = 3$ residual blocks (ResBlock in Figure 4.2) and $R = 2$ repeats for a total of 6 blocks. Secondly, we employ an initial 1D convolutional layer with kernel size 5 and stride 2 instead of a bottleneck convolutional layer. Thirdly, BN is used instead of global layer normalization [53] and depth-wise convolutions in ResBlocks have kernel size of 5. A final linear layer with output size C is used to derive the class logits. A sigmoid activation function is used for DDD ($C = 1$) while softmax is used for multi-KWS experiments ($C = 35$ posteriors, corresponding to the number of keywords in GSCv2, both original and our augmented version). The total number of parameters is 131k. We use 64-dimensional LFBEs as input features extracted with a Short-Time Fourier Transform (STFT) 25 ms window and 10 ms stride. The model is trained using cross-entropy loss on segments of 117 frames, which is the receptive field of the model. If the input length is less than 117 frames, zero padding is employed. During testing, we employ max pooling if the input feature sequence is longer than the receptive field producing a single prediction for the whole sequence. For regularization, we apply SpecAugment [124] independently on both reference and input mixture LFBE features after the initial BN layers. We use the Adam algorithm [64] for optimization and tune the learning rate, weight decay and SpecAugment hyper-parameters for each experiment using the validation split. Each model is trained for a maximum of 200 epochs with 10 epochs early stopping. We use a batch size of 256 and 1200 for the multi-KWS and DDD experiments, respectively. All models are trained on both playback (where reference is available) and non-playback conditions.

Since both KWS and DDD models employ LFBEs features in input, in our experiments we perform the proposed on-the-fly data augmentation strategy described in Section 4.2.2 in the STFT domain, prior to taking the magnitude and apply the Mel filterbank transform. This leads to a minor training speedup. The time-shift is applied on STFT frames and is sampled from $U(15, 20)$ frames. The interferer and target signals are mixed such that the SIR falls in $U(-20, 3)$dB. Note that such rather extreme frame-wise shifting is necessary to simulate the non-deterministic delay in the device playback and input audio pipeline, mainly due to input/output audio software buffers. This delay leads to substantial misalignment between the reference $\boldsymbol{r}$ and corresponding interferer $\boldsymbol{n} = \Gamma(\boldsymbol{r})$ and must be accounted for during training so the model learns to compensate for it.

4.4.2 Device-Directed Speech Detection

In this section we present and discuss our DDD experiments on the real world dataset described in Section 4.3.2 focusing on TTS playback conditions. We use false reject rate (FRR) and false accept rate (FAR) as our metrics and report FRR at two fixed FAR values (non-playback and playback). The FAR values are chosen on the development set according to customer feedback and are redacted in this document due to privacy reasons. As our goal is to obtain a practical on-device streaming DDD classifier, we also report the floating point operations (FLOPs) per output prediction in both playback and non-playback conditions.

In Table 4.1 (upper panel) we report the results for adding simulated interferer signals while training a standard TCN DDD classifier (Baseline) with the architecture explained in Section 4.4.1. Here, we compare the strategies of using a test and dev set matched (default Alexa voice) versus unmatched (Matthew voice) TTS model during training. As expected, adding simulated playback with a matched TTS model (+ Alexa TTS) significantly improves performance especially in playback. On the contrary if an unmatched TTS model is used in training, very marginal improvement is observed in playback while in non-playback conditions FRR degrade significantly. This suggests that the model is learning to ignore the TTS playback mainly based on the “identity" and, to a less extent, perceived gender of the TTS speech. The FRR in non-playback increases for the second model because his perceived gender is male, and the test set is composed mainly by male speakers (while the Alexa TTS voice perceived gender is female). This biases the model to be more prone to consider male speakers as TTS and reject them.

In the second panel, we study the effect of extending the classifier by simply concatenating the reference channel features at the first layer, after BN, as explained in Section 4.2.1. This model requires slightly more FLOPs than the standard Baseline TCN classifier. As our training dataset lacks playback conditions (see Section 4.3.2) also here we resort to simulation using both matched and unmatched TTS voices. However we also study the effect of the data augmentation strategy outlined in Section 4.2.2 which can be leveraged now since the model (iAEC-C) has access to the reference.

The proposed data augmentation strategy alone (iAEC-C augm) is able to improve FRR in both conditions despite the model is trained with no TTS data but only with legit user queries used both for targets and playback roles. Adding simulated matched and, to a less extent, even unmatched TTS interferer/reference data further improves the performance in both conditions. In the Matthew TTS case, the proposed method helps fighting potential biases in the training data and prevent overfitting one particular voice/gender. On the other hand if the iAEC-C model is trained only with matched TTS simulated data and no data augmentation strategy (- *augm* + Alexa TTS) we observe a degradation in performance. This hints that the model with reference access is more likely to overfit the simulated interferer acoustic characteristics in the training set and not generalize well to real-world environments despite our simulation efforts. More complex simulation pipelines may be able to mitigate this issue but have their drawbacks (e.g. lots of hand-tuning). As smart-home assistants offer more voice options, the proposed data augmentation allows to avoid retraining the DDD classifier for each new TTS voice and reduces the risk of rejecting speakers whose voices are similar to the TTS model.

Table 4.1: FRR at fixed FAR (redacted) for non-playback and playback (TTS playing) conditions. We study different training strategies using the default Alexa voice and an alternative TTS voice named Matthew. Intra-dataset mixing (intramix) refers to the on-the-fly mixing strategy outlined in Section 4.2.2.

Model	**Non-Playback**		**Playback**	
	FRR@FAR	FLOPs	FRR@FAR	FLOPs
Baseline	0.189	242k	0.348	242k
+ Alexa TTS	0.187	–	0.241	–
+ Matthew TTS	0.202	–	0.339	–
iAEC-C (augm)	0.188	283k	0.258	283k
+ Alexa TTS	**0.185**	–	**0.227**	–
+ Matthew TTS	0.187	–	0.256	–
- augm + Alexa TTS	0.193	–	0.299	–

The top panel of Table 4.2 investigates the effect of concatenating at deeper residual blocks for iAEC-C from the 1st (D1, same as in Table 4.1) to 3rd (D3) blocks (see Figure 4.2). All models are trained using the data-augmentation (*augm*) strategy outlined in Section 4.2.2 with no simulated playback TTS data. Performance improves with deeper layers up to D2, as the encoder receptive field (FOV) surpasses the maximum offset between reference and interferer signals observed on real-world collected audio, around 200 ms, as said, mainly due to the output and input software audio buffers. On the other hand, also computation increases with the depth at which concatenation is performed. We can see that concatenation at D2 provides the best trade-off between playback FRR, non-playback FRR, and FLOPs.

Table 4.2: FRR at fixed FAR (redacted) using iAEC-M and iAEC-C with concatenation at different layers in the TCN model.

Model		**Non-Playback**		**Playback**	
	FOV	FRR@FAR	FLOPs	FRR@FAR	FLOPs
iAEC-C (input)	5	0.188	283k	0.258	283k
iAEC-C (D1)	12	0.183	336k	0.178	336k
iAEC-C (D2)	28	0.184	369k	0.165	369k
iAEC-C (D3)	60	0.183	402k	0.168	402k
iAEC-M (D2)	28	**0.181**	242k	**0.150**	367k

The bottom panel of Table 4.2 presents the mask-based approach described in Section 4.2.1. Masking is applied at the second residual block (D2) as in iAEC-C and shows a further performance improvement on both playback and non-playback conditions. Regarding computational efficiency at D2, iAEC-M requires slightly more FLOPs in playback mode than iAEC-C, but significantly less FLOPs in non-playback mode, as the reference branch is dropped and the architecture becomes equal to a standard classifier. Since this model is ran continuously, playback conditions account for a very small fraction of total inference time and thus iAEC-M leads to the best performance/FLOPs trade-off.

4.4.3 Multi Keyword Spotting

In Table 4.3 we report our results on the augmented GSCv2 dataset described in Section 4.3.1. We report keyword-detection accuracy over the 35 possible GSCv2 keywords for the 3 different test sets in our augmented version: original GSCv2 (*Non-Playback*), mixed with music (*Playback Music*) and with TTS (*Playback TTS*).

As with the DDD experiments, we use a standard TCN classifier without access to the reference as our *Baseline*. We also consider a joint model (*+nAEC*) comprised of the state-of-the-art neural AEC model from [105], designed for edge-devices applications, and the *Baseline* TCN classifier: the output of the nAEC is directly fed to the classifier in cascade. Moreover we compare with an STFT-based normalized least mean squares (NLMS) AEC algorithm with 32 taps, step size $\mu = 0.5$, 512 and 128 STFT window and hop with square-root Hann window. Pyroomacoustics [125] was used for the implementation.

For this system we use a weighted loss consisting of a spectral loss term as in [105] and a KWS loss term instead of an ASR on as in [105]. The two cascaded models nAEC and TCN are then jointly trained. As another baseline, we include the performance for MatchBoxNet-3x2x64 [126] (MatchBN) using the official implementation from NeMo toolkit [127]. We also report results for three different training strategies for the models with access to the reference (thus including [105]). In *orcl* we train the model with access to the oracle reference, interferer and target signals, e.g. the nAEC model is trained to estimate the target and is given in input the reference corresponding to the interferer together with the mixture signal. This is a best-case scenario, possible because this dataset is fully synthetic. Here there is no mismatch between training and test data, a rather ideal condition which e.g. will prevent the degradation observed in DDD experiments in Table 4.1 last row due to mismatched simulated training and real-world interferer acoustic conditions. The *augm* denotes the augmentation strategy described in Section 4.2.2 where, in training, we generate fake playback mixture signals by mixing original GSCv2 examples. Each example is randomly given the role of interferer/reference or target, without adding any simulated playback TTS or music. Finally the strategy denoted *both* denotes a combination of these two: in training one example is either generated on-the-fly with *augm* or comes from *orcl* with 50% probability.

As expected, we can observe that for both *Baseline* and *MatchBN* playback performance degrades significantly, especially during TTS playback conditions, which can confuse the model. Regarding the models with access to the reference (Baseline +nAEC, iAEC-C and iAEC-M), we can see that, in the best case scenario of matched training and testing (*orcl*), all models significantly improve performance over the Baseline classifier especially in playback conditions. The improvement in non-playback is due to the fact that the MatchBN and Baseline models are prone to reject a valid keyword as TTS playback, as they have no access to the reference. If only the *augm* strategy is used in training the performance degrades compared to the ideal *orcl* data but still affords improvement over models with no reference access. On the other hand, as explained, in many real-world applications the reference signals can be difficult to obtain or may be biased (e.g. all examples with playback belongs to one TTS model). Combining both strategies yields the best performance overall for all models (including nAEC) even surpassing *orcl*. The NLMS AEC is very effective with the music interferer but struggles with TTS. On the other hand it is extremely light computationally and thus future work could explore combinations of this approach and the proposed one.

Overall, our proposed iAEC approaches perform competitively with *+nAEC* but uses two order of magnitude fewer FLOPs for each prediction. This makes the proposed methods more suitable for always-on low-resource applications.

Table 4.3: Accuracy and FLOPs on GSCv2 for different models and training strategies (see Section 4.4.3). Metrics are reported both for original data (*Non-Playback*) and our simulated playback corpus, separately for *TTS* and *Music* playback conditions.

Model		**Non-Playback**		**Playback**		
		Acc %	FLOPs	Acc % (Music)	Acc % (TTS)	FLOPs
MatchBN [126]		94.47	185k	61.84	36.78	185k
Baseline		93.35	242k	75.01	61.71	242k
+ NLMS AEC		93.77	242k	78.95	63.13	243k
+nAEC	orcl	94.06	242k	82.87	82.75	15M
	augm	93.82	-	75.04	72.21	-
	both	94.46	-	83.55	**83.81**	-
iAEC-C	orcl	94.52	283k	82.60	80.79	283k
	augm	94.56	-	77.91	77.52	-
	both	94.74	-	83.54	82.93	-
iAEC-M	orcl	94.67	242k	83.87	82.47	367k
	augm	94.49	-	78.21	77.01	-
	both	**94.97**	-	**84.22**	83.79	-

4.5 Conclusions & Future Work

In this Chapter we introduced the two related problems of keyword spotting KWS and device-directed speech detection DDD. Both these tasks are quintessential for voice-enabled human-machine interaction, with DDD especially promising (albeit extremely challenging) as a step towards a more natural interaction. As we have seen these two tasks have to be run on-device as they absolve the role of a gateway to the server-side and more complex back-end tasks such as ASR. This fact however is also what makes such tasks particularly challenging: they have to operate reliably, in a continuous manner, with low-latency and have to be computationally lightweight.

In our work on implicit acoustic echo cancellation (iAEC), in particular we addressed the problem of boosting KWS and DDD classifiers performance on edge-devices during device playback without degrading non-playback performance. This work also introduced DDD as a streamable, on device task, a considerable feat due to its open vocabulary nature. In our proposed framework the DDD and KWS classifiers leverage the known playback signal (reference signal) to ignore the return “echoed" playback signal (interferer signal) captured by the device microphones together with the user speech.

We explored two strategies for feeding the reference signal to our models and found the use of a latent-space masking approach particularly suited for our KWS and DDD tasks, as it brought significant performance improvements in device playback conditions. On the KWS task the proposed method obtains comparable performance with a state-of-the-art neural AEC method but with much less computational requirements.

As an additional contribution, we devised an effective data augmentation strategy that is able to further boost performance of neural AEC and iAEC models, allows to train such models on examples with no playback interferer and helps reducing bias in the training data.

Future work could explore other tasks such as ASR or speech to speech translation and scenarios beyond device playback where oracle double-talk detection is not available, such as teleconferencing.

Chapter 5

Speaker Counting, Voice Activity and Overlapped Speech Detection

Context

The work in this Chapter was presented at Interspeech 2020 [99] and also as an extension to Computer Speech and Language in 2021 [128]. It was done in collaboration with Emmanuel Vincent from Université de Lorraine and Maurizio Omologo from Fondazione Bruno Kessler. The main idea came in 2019 during the JSALT 2019 workshop with Emmanuel suggesting the use of spatial features for speaker counting.

5.1 A Brief Historical Overview

VAD is an indispensable task in most speech processing applications. As KWS and DDD it absolves as a “gateway” for downstream processes. In fact these latter could be designed only to handle speech (and not long silences) and/or are too computationally heavy to be ran continuously, or again, to save bandwidth if VAD is performed on edge-devices. In fact, the first VAD approaches were actually carried out with the main goal of reducing the bandwidth in telecommunications. The advent of the digital age in the 70s, opened up exciting prospects towards ASR and, at the same time, digital signal processing VAD algorithms started to be devised, initially relying on very simple features such as the energy of the signal or the zero crossing rate[129]. In the following decades the research towards robust VAD was mostly focused on devising more reliable hand-crafted features with the goal of improving detection performance in noisy-reverberant scenarios. These included harmonic features [130] or spectral shape [131] or assumptions such as stationary noise [132]. In these early approaches, the speech/non-speech decision was made with heuristic rules e.g. a series of if-this-the-that rules such as the current frame is speech when an hand-tuned threshold value for the energy is exceeded. One of the most effective approaches following this paradigm was the system proposed by Ramirez et al. [133] which relies on a sub-band and on long-term (multiple-frames) averaging for more reliable estimation. It also incorporates a Wiener filter for denoising the signal prior to the VAD decision. The noise statistics for the Wiener filter are taken from the first frames of the input signal as they are assumed to be speech-free.

Towards the turn of the millennia, more statistically-principled approaches to the VAD problem started to emerge. One of the first works in this direction was performed by Sohn et al. [134] in which many advanced concepts were proposed such as an hidden Markov model (HMM) hang-over scheme and a decision rule based on the likelihood-ratio. Their method relied on the assumption that the STFT bins values could be modelled by complex gaussian distributions with different variance for the estimated noise the estimated speech bins. Their approach was able to outperform significantly the state-of-the-art at the time VAD used in the ITU standard G.729B [135] for telecommunications audio compression purposes. This general framework was improved over the years. For example, Shin et al. [136] proposed a follow-up work where the complex gaussian distribution is instead replaced with a generalized gamma distribution, which has more modeling capacity. Chang et al. [137] instead proposed to use an ensemble of different statistical models.

In subsequent works [138, 139], VAD systems started to incorporate more principled data-driven approaches such as gaussian mixture model (GMM) [140, 141] or support vector machine (SVM)[138, 139] as more computing power got available even on edge-devices. This eventually led to the adoption of DNN-based methods, which are now the de-facto mainstream approach when sufficient robustness is needed e.g. when dealing with far-field speech. One of the very first works on neural VAD was done by Eyben et al. [142] in 2013. The authors used an LSTM and their results showed a remarkable performance increase with respect to previous methods such [133, 134]. More up-to-date work [143] focused on real-time extremely constrained VAD applications by using an multi-layer perceptron (MLP)-based method, while [144] proposed a fully end-to-end hybrid CNN-LSTM network. Some works such as [145] and [146] also explored the use of multi-channel features for VAD. Vesperini et al. [145] compared several different architectures for this purpose while [146] proposed an end-to-end system for joint localization and VAD. It is also worth mentioning the very effective Pyannote VAD model by Bredin et al. [147] which relies on end-to-end learned features by using a SincNet front-end [148].

The research towards reliable OSD instead is more recent than the one for VAD, but still spans more than one decade, with again the first systems relying on handcrafted features and classical machine-learning approaches. Historically, the main factor that led the development of OSD algorithms was the need to improve diarization for ASR speaker adaptation. Most of these early studies focused on Gaussian Mixture Model (GMM) or Hidden Markov Model (HMM) based classifiers [149–153] with the exception of [154] who showed a Long-Short Term Memory (LSTM) neural network to outperform a GMM-HMM system. [149], [150], and [152] reported a substantial reduction of the Diarization Error Rate (DER) on the AMI meeting corpus [155] by removing overlapped speech segments from the segment clustering phase and performing overlap attribution afterwards.

When multiple microphone channels are available, speaker counting can be performed by clustering inter-channel features [156, 157] or explicitly localizing the speakers in space [158, 159], both in the single-array and multiple-array scenarios. Single-channel speaker counting is more challenging, with early works focusing on handcrafted features such as the modulation index [160], the mean and variance of the 7th Mel filter [161] or the cosine similarity between Mel Frequency Cepstrum Coefficient (MFCC) feature vectors along with pitch [162]. More recently, [163] estimated the number of speakers by computing the distance between the mixture and a reference single-speaker utterance in the magnitude spectral domain.

CountNet [164] marked a significant departure from these previous works by showing that a neural network can be trained to perform speaker counting without relying on handcrafted features, and it can even outperform humans. [165] also showed that a neural network based speaker counting algorithm can defeat human ability especially when more than three speakers are active. [166] took a different direction: they trained a neural network to perform joint speaker counting, speech recognition and speaker identification in a fully end-to-end fashion. In all these works, synthetic mixtures are employed for both training and testing and, crucially, the datasets are designed with balanced proportions of single-speaker speech, two-speaker overlapped speech, three-speaker overlapped speech, and so on. This does not match the characteristics of real-world datasets where single-speaker speech is more frequent than two-speaker overlapped speech, which is itself much more frequent than three-speaker overlapped speech.

Regarding OSD, [167] and [168] recently showed that deep neural networks significantly outperform classical machine-learning approaches for this task too. Notably, [168] evaluated four network architectures for joint VAD and OSD (VAD+OSD): a feedforward network, a 2-D convolutional network, a recurrent LSTM network and a hybrid 2-D convolutional-LSTM network. They showed that these approaches surpass a baseline GMM-based method on both synthetic data and AMI distant-speech data, that the LSTM-based approach performs best, and that it significantly improves diarization results. More recently [169] and [170] reported impressive OSD performance in near-field conditions, with

[170] reporting up to 20% relative Diarization Error Rate (DER) reduction on the AMI headset mix. In another vein, [171] addressed VAD+OSD by employing simple classifiers on top of pre-trained x-vector speaker embeddings [172] and evaluated them on synthetic data corrupted by noise and artificial reverberation.

5.2 Overlapped Speech Detection and Counting Framework

In our work [128], we proposed to treat supervised VAD, OSD, VAD+OSD, and speaker counting in a unified way, as special instances of a general OSDC task. This task can be formulated as a multi-class supervised sequence labeling problem, with a different number of classes for VAD, OSD, joint VAD+OSD, and speaker counting.

We consider a parametric model $\mathcal{F}(\mathbf{X};\boldsymbol{\theta})$ which takes as input a sequence of frame-level feature vectors $\mathbf{X} = \{\mathbf{x_1}, \mathbf{x_2}, \dots, \mathbf{x_m}\}$ and outputs a sequence of class posterior probabilities. We assume that the model may perform internal subsampling, i.e., one output frame is provided every K input frames. This is because frame-level estimation is unnecessary for most speech segmentation applications and, by employing subsampling operations, the computational burden can be reduced.

In the supervised setting, we are given the ground-truth class label sequence $\mathbf{y} = \{y_1, y_2, \dots, y_l\}$ of length $l \leq m$, and we wish to estimate the optimal model parameters $\widehat{\boldsymbol{\theta}}$ according to a certain criterion. As in this work we focus on neural approaches, the optimal model parameters are estimated on a suitable training set composed of N pairs of input feature sequences and corresponding class label sequences $\mathbf{T} = \{(\mathbf{X_1}, \mathbf{y_1}), \dots (\mathbf{X_N}, \mathbf{y_N})\}$ by using Stochastic Gradient Descent (SGD) to minimize the cross-entropy loss between the estimated frame-level posterior probabilities and the true class distribution.

In this framework, VAD and OSD can be treated either separately as binary classification tasks (speech vs. non-speech, overlap vs. non-overlap), or jointly as a three-class (non-speech, single speaker, overlapped speech) problem. Similarly, speaker counting can be formulated as an C-class classification task where C is equal to the maximum possible number of overlapping speakers plus one. While this approach is not the only one for supervised speaker counting, it has been found to be the most effective [164], provided the maximum possible number of concurrent speakers is known.

5.3 Proposed Neural Architectures for OSDC

We studied four neural network architectures for tackling the OSDC task.

5.3.1 Long-Short Term Memory (LSTM)

The first one is the best neural network for joint VAD+OSD among the ones examined by [168] which, to our knowledge and with the exception of our preliminary work [99], achieves the best reported performance on AMI single-channel distant-speech data.

It consists of a unidirectional LSTM layer with a hidden size of 512 neurons, followed by 3 dense layers with 1024, 512 and 256 neurons, respectively. A final $256 \times N$ pointwise convolutional layer along with softmax is used to output the probability of each frame belonging to one of the N classes (e.g., $N = 3$ for VAD+OSD). This network features a total of 2 million parameters and generates one output vector for every input frame given a context of 11 frames (current frame plus 5 past and 5 future frames).

As the original architecture lacked any normalization technique, in our experiments we added batch normalization [120] before each dense layer activation as well as layer normalization [173] on the input features. This, coupled with data-augmentation, allows us to improve performance over the original network as it will be shown in Section 5.6.5.

5.3.2 Hybrid Convolutional-Recurrent Neural Network

We also consider the best CountNet architecture among the 5 different networks compared by [164]. This network is a hybrid Convolutional-Recurrent Neural Network (CRNN), composed of a 2-D Convolutional Neural Network (CNN) block followed by an RNN block. The main idea behind this architecture is that the CNN extracts a local representation of the input features while the RNN deals with long-term temporal modeling, thus combining the advantages of both CNNs and RNNs.

Input features of shape $F \times T$ are fed to the CNN which is composed of two blocks, each composed of two 2-D convolutional layers with kernel size 3×3 followed by ReLU activation and a 3×3 max-pooling subsampling operation. A total of 4 convolutional layers is thus employed with 64, 32, 128 and again 64 channels, respectively. Dropout [174] is applied on the output of the CNN and the representation is fed to an LSTM layer with a hidden size of 40. As an LSTM operates on 2-D sequences while the output of the CNN is a 3D tensor with channel, frequency, and time dimensions, a 2-D sequence is obtained by stacking the frequency dimension onto the channel dimension.

[164] performed an additional max-pooling operation on the whole time dimension in order to output a single prediction for the entire input because they aimed to count the maximum number of speakers in the whole sequence. Here, as explained in Section 5.2, we are interested in estimating the number of speakers in each time frame instead so we omit this final pooling layer. In this way, the network generates one output vector for every 6 feature vectors in input. As this architecture also originally lacked any normalization strategy, we added batch normalization after each convolutional layer and layer normalization at the input.

5.3.3 Temporal Convolutional Network

In addition to the above two state-of-the-art architectures, we consider a TCN architecture for the OSDC task. This type of architecture has been shown to achieve state-of-the-art performance in many sequence-related tasks [175] and for source separation [176].

TCNs rely on multiple stacked dilated convolutional layers whose dilation factor increases progressively as depth increases. This makes it possible to greatly expand the receptive field, such that upper layers can have access to long-term contextual information without any pooling operation. This in turn allows TCNs to outperform recurrent models in some tasks [175]. In fact, because they are based only on convolutional operations, TCNs have several benefits with respect to RNNs. First, being feedforward, they are not affected by the vanishing gradient problem which plagues RNNs, as skip-connections and residual connections can be used to backpropagate the gradient unscathed down to the very first layers. Second, in RNNs the information about the past must be contained in the hidden state. This makes it difficult to learn very long-term dependencies as all relevant information about the past must be squeezed into this finite-sized representation. On the contrary, TCNs process the whole sequence and, because no downsampling is performed, the information at all steps is preserved in all layers. Finally, as no recurrent operations are employed, TCNs are significantly faster than recurrent models in both the training and inference phases. However, the fact that the representation is not pooled leads TCNs to have large memory requirements in general, especially if a very wide receptive field is desired.

The architecture we employ here [99] is depicted in Fig. 5.1. It is inspired from MobileNet [177] and Conv-TasNet [176]. Input frame-level feature vectors of size F (e.g., log-Mel filterbanks) are fed to a layer normalization [173] layer followed by an $F \times 64$ 1D pointwise convolutional layer (denoted as *conv 1x1*) and by $R = 3$ blocks of $X = 5$ residual blocks (*res blocks*) with 1D dilated convolutions, where the dilation factor increases in each block as $2^0, 2^1, \ldots, 2^{X-1}$.

Each residual block consists of a 64×128 pointwise convolutional layer followed by batch normalization and activation, a dilated depthwise separable 128×128 convolutional layer (*d-conv*) followed by batch normalization and activation, and another 128×64 pointwise convolution which squeezes the representation back so that it can be summed with the input. We use PReLU [178] as the activation function in all residual blocks and a kernel size of 3 in depthwise dilated convolutions.

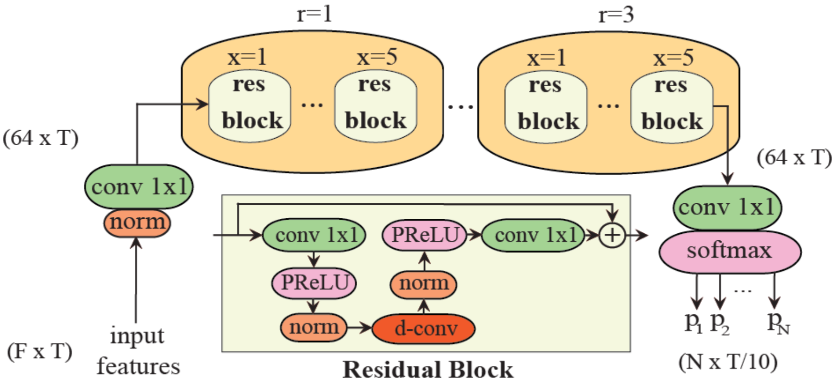


Figure 5.1: Proposed TCN architecture for the OSDC task.

5.3.4 Transformer

Finally, we propose a Transformer-based architecture for OSDC. Transformers, which were originally proposed by [11] for natural language processing applications, are pure attention-based models which have been shown recently to achieve state-of-the-art performance in many speech processing tasks including diarization [179]. They have several advantages over recurrent models, including faster inference speed and better modeling of long-term dependencies. In fact, as they are feed-forward models, the whole sequence is attended at once, eliminating any recurrence and any need for an internal hidden state to keep track of past elements. Onthe contrary, in recurrent architectures the information about the past elements has to be memorized in the internal hidden state, whose size is fixed. For this reason, Transformers exhibit the same advantages as TCNs over RNNs, even if their inherent functioning is significantly different. Similarly to TCNs and while being much faster than RNNs, Transformers also have higher memory requirements, due to the fact that the attention mechanism grows as $\mathcal{O}(n^2)$ in memory with n the length of the input sequence.

Our Transformer-based architecture is depicted in Figure 5.2 and, as it can be seen, has some input and output blocks in common with the previously described TCN network. To counter the quadratical memory growth induced by the attention mechanism, we adopt a concatenate-subsample (*cat-pool*) operation over the input feature vectors. For each frame, we concatenate the feature vectors from C past frames and C future frames with the current one. Afterwards, we subsample this representation on the frame axis by a factor of S. In this way, the information contained in the temporal dimension is effectively transferred to the feature dimension with a resampling factor of C/S the original rate. This concatenated and pooled representation is then fed to a layer normalization layer followed by a pointwise convolutional layer (*conv 1x1*) which shrinks the representation to a predefined size H to reduce the memory requirements of subsequent blocks, allowing us to process longer sequences or, alternatively, to reduce the computational footprint of the model as it will be shown in Section 5.6.4. Sinusoidal positional encoding is added right after this bottleneck convolutional layer and the result is fed to a succession of R Transformer Encoder blocks, each composed of two residual sub-blocks.

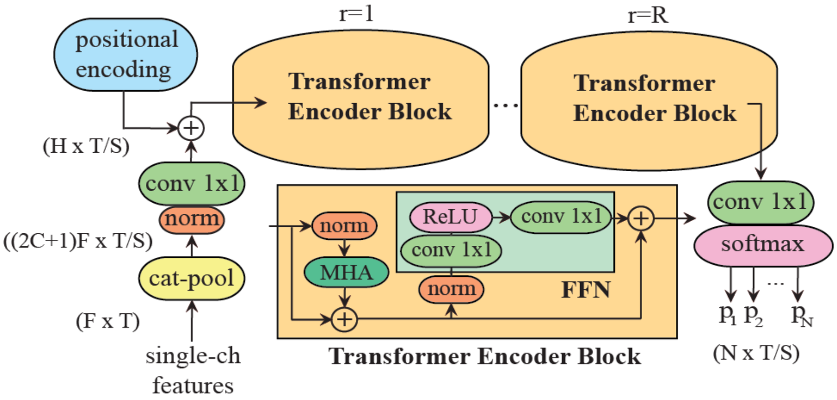


Figure 5.2: Proposed Transformer architecture for the OSDC task.

The structure of each Transformer Encoder block is identical to the one proposed by [11] with the exception that, in our architecture, layer normalization is performed at the beginning of each residual block rather than in the end. Indeed, [180] recently found that this results in better performance as well as faster convergence. The first residual block consists of a normalization layer followed by a Multi-Head Attention (MHA) layer and dropout. The second one consists of a normalization layer followed by a position-wise feedforward neural network (FFN) composed of one dense layer,[1] a ReLU activation followed by dropout, and another dense layer which projects the hidden representation back. As in the TCN model, a final $H \times N$ pointwise convolutional layer followed by softmax is used at the output.

[1] Note that dense layers are equivalent to *conv 1x1*.

5.4 Spatial Features and Feature Fusion Schemes for OSDC

Intuitively, spatial features can help VAD, OSD and speaker counting. For example, OSD and speaker counting can benefit from knowing whether the sound comes from one or more Directions of Arrival (DoAs). VAD can also benefit from spatial features to distinguish speech, which is usually directional, from noise, which can be spatially diffuse.

In fact, as mentioned in Section 5.1, many works have tackled speaker counting by framing it as a localization problem. These works resort to DoA estimation methods based on generalized cross-correlation with phase transform (GCC-PHAT) [181] as in [156, 158], magnitude-squared coherence (MSC) [157] or simple cross-power spectrum [159, 182]. The speaker number is estimated via a direct approach such as in [158] by counting peaks in GCC-PHAT based acoustic maps or by clustering methods, where speaker clusters are identified by iterative grouping of complex-valued time-frequency coefficients [156], magnitude squared coherence feature vectors [157], or DoAs estimated over single-source time-frequency zones [159] or individual time-frequency bins [182].

Recently, a series of works have proven that neural network based localization is more robust than signal-based methods in reverberant and noisy environments. In these works, a neural network is trained to estimate the DoA on a synthetic dataset for which the true position of the sources is known. Input features include GCC-PHAT [183], cosine-sine interchannel phase difference (CSIPD) features [184], the phase spectra of all channels (phasemap) [185], the magnitude and phase spectra [186], or the raw waveform [187].

5.4.1 Signal-based Spatial Features

In this work, for what concerns signal-based spatial features, we explore the interchannel phase difference (IPD) and CSIPD, as they have been shown in the aforementioned works to work well in reverberant and noisy environments. In particular, our choice of IPD instead of phasemap is justified by the fact that, both in AMI and CHiME-6, microphones are close to each other and thus some microphone pairs can be discarded as they do not add much spatial diversity at 16 kHz. On AMI, we consider only those pairs of microphones with maximal distance from each other, i.e., the 4 pairs formed by opposite microphones in each circular array instead of all 28 possible pairs. On CHiME-6, due to the asymmetrical placement of microphones in Kinect devices, we consider the 3 pairs formed by channels 1 and 4, channels 2 and 4, and channels 3 and 4. The IPD or CSIPD features of all pairs are then concatenated together over the frequency dimension. Thus, in these contexts, using interchannel features allows us to reduce the feature size with respect to the phasemap and hence save computational resources with practically no loss in spatial information.

IPD and CSIPD features are tightly related and derive from the phase spectrum. Denoting by $x_i(n, f)$ and $x_j(n, f)$ the STFT of the i-th and j-th microphone signals, where n and f are respectively the frame and frequency index, the IPD $\phi_{i,j}(n, f)$ between channel i and j is given by

$$\phi_{i,j}(n, f) = \angle x_i(n, f) - \angle x_j(n, f), \tag{5.1}$$

where $\angle(.)$ is the function returning the phase from the input complex value. The IPD feature vector in time frame n is then defined as

$$\mathbf{IPD}(n) = \left[\phi_{i,j}(n, 0), \phi_{i,j}(n, 1), \ldots, \phi_{i,j}\left(n, F/2\right)\right]^T, \tag{5.2}$$

with F the FFT size. IPD features are depicted in Figure 5.3.

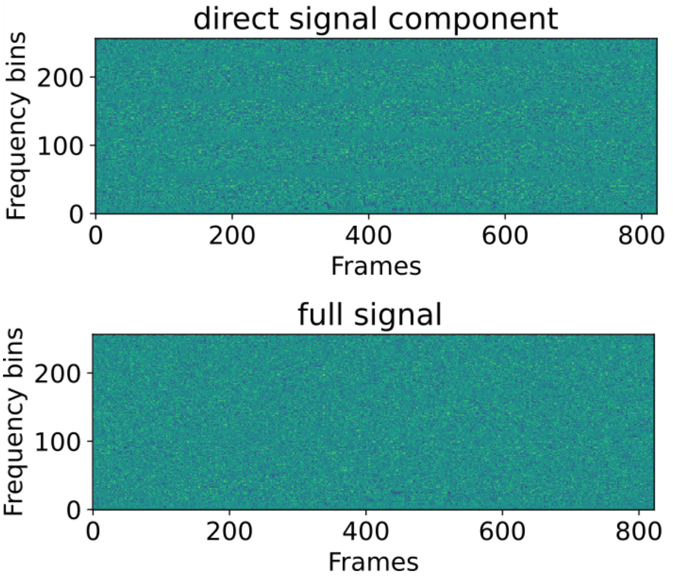


Figure 5.3: Overview of IPD features, extracted with 16 ms STFT window with 4 ms stride from an utterance in the synthetic dataset described in Section 5.5. We used the two microphones at the edges of the linear array. Top: using only the direct anechoic signal component. Bottom: using in input the full signal.

The CSIPD feature vector in time frame n can be obtained directly from the

IPD feature vector and is another way of encoding the information contained in it by using its cosine and sine values:

$$\mathbf{CSIPD}(n) = \left[\cos\phi_{i,j}(n,0), \sin\phi_{i,j}(n,0), \ldots, \sin\phi_{i,j}\left(n, F/2\right)\right]^T . \tag{5.3}$$

An important property of CSIPD is that the GCC-PHAT angular spectrum for a given microphone pair (or the SRP-PHAT spectrum when there are 3 or more microphones and all pairs are considered) can be expressed as a linear transformation of the CSIPD feature vector [188]. When these features are to be input to a neural network model, there is therefore no benefit in using the GCC-PHAT or SRP-PHAT angular spectra as inputs instead, since this linear transformation can be learned by the neural network itself. This was confirmed by our experiments, so we do not report results obtained with GCC-PHAT or SRP-PHAT features in the following. CSIPD features are depicted in Figure 5.4. We can see how they exhibit much more evident structure with respect to IPD.

5.4.2 Neural Network-based Localization Features

As an alternative, we also consider the strategy of training a neural network to estimate the DoAs of multiple overlapped speakers on a suitable synthetic dataset for which the true DoAs are known. The embeddings extracted by some intermediate layer of this network can then be used as “higher-level”, possibly more robust spatial features to be employed in the OSDC system. In this work, we adopt the multi-speaker localization method of [185], where the space of DoAs is discretized and the neural network is trained to estimate the posterior probability that a speaker is active for each discrete DoA by minimizing the sum of binary cross-entropies across all discrete DoAs. Binary cross-entropy is used as the cost function since multiple concurrent speakers with different DoAs can be active at the same time.

In detail, even for localization, we use the networks outlined in Section 5.3 by modifying the output layer which is replaced with mean pooling over the sequence dimension and a new linear layer with output size D followed by sigmoid activation, where D is the number of discrete DoAs considered. The network representation before the mean pooling operation is then employed as a spatial feature vector for OSDC systems.

One advantage of neural network-based features over signal-based features is that joint fine-tuning of the two networks can be performed, thus optimizing the localization feature extraction network for OSDC applications. However, it must be noted that the computational footprint significantly increases by using neural network based features. Also, the fact that true source DoAs are needed for training necessitates the use of a synthetic training dataset, which

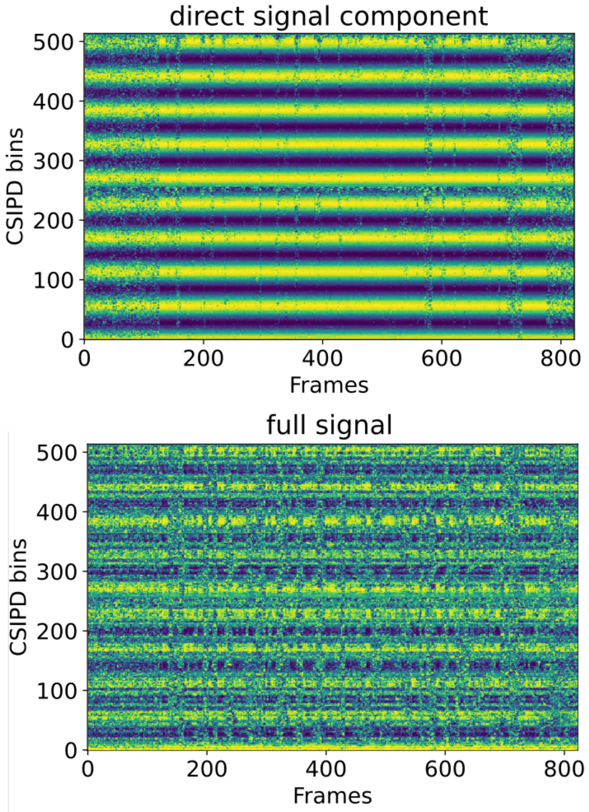


Figure 5.4: Overview of CSIPD features, extracted with 16 ms STFT window with 4 ms stride from an utterance in the synthetic dataset described in Section 5.5. We used the two microphones at the edges of the linear array. Top: using only the direct anechoic signal component. Bottom: using in input the full signal.

can be mismatched with real-world data.

5.4.3 Fusion schemes

Spatial features are not sufficient for reliable OSDC when used alone. For example, directional noise sources may sometimes be confused with speech, or concurrent speakers can have the same DoA. They must hence be combined with single-channel spectral features, such as log-Mel spectra. We consider two different fusion schemes for this combination, which we call early and late fusion.

These fusion schemes are illustrated in Figure 5.5 for the Transformer-based network. In early fusion, the two features are stacked together in the very first layer of the neural network. Layer normalization on spatial features is performed separately prior to concatenation. In late fusion, after layer normalization, the spatial features are injected before each Transformer Encoder Block (*TE Block*), using Feature-wise Linear Modulation FiLM [189]. In this way, each block of the architecture can focus on a different aspect of the input spatial features since they are available even in deeper layers. As the spatial and single-channel features are concatenated together in early fusion, they must have same temporal length. Thus, for proposed Transformer network we employ the same cat-pool operation also on spatial features prior to concatenation. The same argument applies also for late fusion where instead spatial features are used to modulate activations at multiple layers.

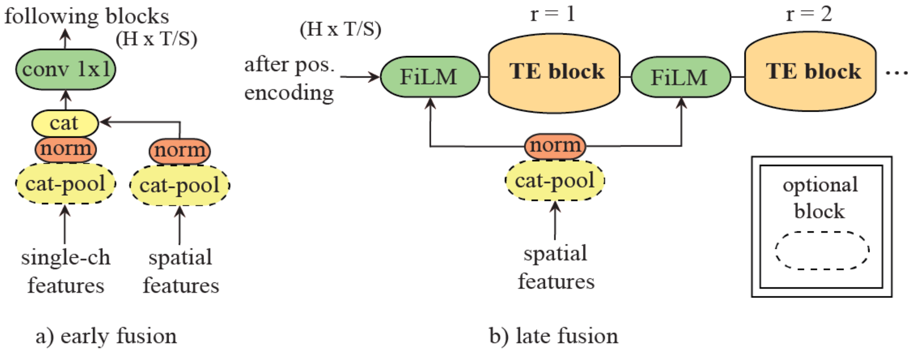


Figure 5.5: Fusion strategies for single-channel features and spatial features for the proposed Transformer architecture: a) early fusion, b) late fusion. TE stands for Transformer Encoder.

5.5 Datasets

We conduct experiments on two real-world multi-microphone datasets: AMI and CHiME-6. Moreover, we also use a synthetic dataset to further study, in a controlled situation, the use of spatial features to improve the performance of OSDC systems. CHiME-6 has already been described in Chapter 3, Section 3.3.2 and we refer the reader to such section.

5.5.1 Synthetic Dataset

We simulate multi-speaker mixtures captured by a single microphone array. Clean speech utterances are taken from Librispeech [190] *train-clean-100* for training, *dev-clean* for validation, and *test-clean* for test. The Montreal Forced Aligner (MFA) [191] is used to split these original Librispeech utterances in order to obtain shorter "sub-utterances" for each speaker. This splitting is performed whenever pauses of more than 150 ms are encountered. MFA is also used, in parallel, to obtain ground truth word-level speaker activity. For each mixture, we sample from 1 to 4 different speakers, and, for each speaker one clean speech sub-utterance is sampled. The starting time of each speaker sub-utterance is sampled independently from an exponential distribution. In this way, by varying the decay rate parameter, the amount of overlap between the speakers and the amount of silence can be controlled. A different acoustic scenario is sampled for each mixture. We simulate a rectangular room whose size is varied between 10 and 60 m^2. The position of each speaker is chosen randomly inside the room but with some constraints. Namely, the speakers cannot be less than 0.5 m from each other and from the walls. We consider a 4-microphone linear array placed randomly with respect to the walls, whose height with respect to the floor can vary between 1.7 and 2 m and whose distance to the closest wall is larger than a minimal distance which is varied between 10 and 30 cm. We use the gpuRIR [192] toolkit for room simulation with a T60 reverberation time uniformly sampled between 0.2 and 0.6 s. Anechoic noise from [193] is also employed to make the dataset more realistic. The positions of noise sources inside the room are selected with the same criteria as the speakers' ones. The whole synthetic dataset consists of a total of 10 k mixtures ($\sim$ 23 hours) for training, 2 k for validation and 2 k for test ($\sim$ 4.6 hours).

5.5.2 AMI

The AMI Corpus [194] is over 100 h of meeting recordings. Each meeting has been recorded by a variety of devices including cameras, microphone arrays, and per-speaker headset and lapel microphones and has from 3 to 5 participants.

Ground truth speaker activity was obtained by human annotators from close-talk speaker-worn microphones while distant speech was recorded by two 8-microphone circular arrays, each with a 10 cm diameter: one placed at the end and another at the centre of the meeting table used by the participants.

5.6 Experimental Analysis

In the following, we evaluate the neural architectures in Section 5.3 and the spatial features and feature fusion schemes in Section 5.4 on the datasets described in Section 5.5. Firstly, in Section 5.6.1, we define and motivate the chosen performance metric. In Section 5.6.2, we outline the training and testing procedure adopted in our experiments and, in Section 5.6.3, we highlight the impact of different choices of hyperparameters and single-channel input features for the Transformer-based architecture. Then, in Section 5.6.4, we provide an analysis of the computational footprint of the four considered neural architectures when applied to single-channel data and, in Section 5.6.5, we report their OSDC performance on AMI and CHiME-6. Finally, in Section 5.6.6, we assess the impact of spatial features on the best single-channel system: we explore different spatial features, fusion schemes and number of microphone pairs, and evaluate the results on AMI, CHiME-6 and the proposed synthetic dataset.

5.6.1 Evaluation Metric

On real-world data, VAD, OSD and speaker counting tasks are affected by class imbalance. This imbalance, which arises from intrinsic characteristics of human conversations, can be more or less severe depending on the context. This can be seen in Table 5.1, which reports the class statistics on AMI and CHiME-6 for the counting task.[2] Due to its informal, “cocktail-party” scenario, the CHiME-6 dataset exhibits a slightly higher proportion of overlapped speech than the AMI dataset, which consists of meetings. Nevertheless, in both datasets, the proportion of 4-speaker and 3-speaker overlap is very small. The imbalance is less severe for VAD and OSD tasks but, even for these, the choice of the evaluation metric can be crucial.

We argue that metrics such as accuracy and precision-recall, as used respectively by [168] and by [169] and [170], do not provide a fair evaluation of OSDC algorithms on real-world data due to this fundamental imbalance. For example, concerning OSD on the AMI evaluation set, an accuracy of 83.7% can be reached by labeling all the material as no-overlap. As it has been observed by [99, Table 5], this leads to small accuracy differences even for classifiers

[2] We disregard the 5-speaker overlap class on AMI since it does not occur in practice.

Table 5.1: Frame-level class frequency (%) for the speaker counting task on the AMI and CHiME-6 development and evaluation sets.

Class frequency		0-spk	1-spk	2-spk	3-spk	4-spk
AMI	dev	15.87	67.17	13.95	2.59	0.42
	eval	15.12	68.39	12.63	3.09	0.76
CHiME-6	dev	24.05	54.25	17.74	3.49	0.47
	eval	33.47	51.52	12.03	2.46	0.51

with drastically different performance. In this scenario, precision and recall are a better choice than accuracy. However, similarly to accuracy, their value depends on the choice of the detection threshold which can be application-specific (e.g., a different threshold for diarization and speech recognition is often desirable). This does not allow for a fair comparison between different OSDC algorithms.

For these reasons, we propose the use of Average Precision (AP) metric which summarizes the precision-recall curve and is widely used, for example, in object segmentation [195], information retrieval [196] and other tasks exhibiting strong class imbalance. It can be obtained from precision P and recall R at the k-th threshold as:

$$AP = \sum_{k=1}^{M} (R_k - R_{k-1}) P_k, \tag{5.4}$$

where M is the total number of unique thresholds considered. The number of elements in this set is upper bounded by the number of unique elements in the classifier output probabilities vector. In this work we use each time the maximum number of possible thresholds to compute AP. As it can be seen, AP has the advantage that it does not depend on a particular threshold, making it both more robust to imbalanced data and more suitable for comparison purposes. In all experiments, AP scores are computed on 10 ms time frames.[3]

AP in Equation (5.4) is suitable only for binary classification tasks such as VAD and OSD. However, it can be extended easily to multi-class classification problems such as Speaker Counting with a leave-one-out classification strategy: e.g. AP for the 1-spk class can be obtained by considering an equivalent binary task where the true positives are the frames correctly classified as 1-speaker and the true negatives are the frames classified as silence (0-spk), 2-speaker (2-spk), 3-speaker (3-spk), or 4-speaker (4-spk). In a similar way one can compute VAD and OSD AP from a neural network trained to perform Speaker Counting or VAD+OSD. For example, for a Speaker counting algorithm, VAD predictions

[3] The sequence output by the Transformer model is stretched by a factor of S, in order for the number of input and output frames to be equal, similarly to the other models.

can be obtained by summing the probabilities for the classes with at least one speaker: 1-spk, 2-spk, 3-spk, and 4-spk, thus obtaining the total probability of speech; OSD by summing classes with at least 2 speakers: 2-spk, 3-spk, and 4-spk.

Unless stated otherwise, in each Table, we highlight in bold font the best result and the ones which are statistically equivalent to it (if any) with $p = 0.001$. Because we found the distribution of the AP metric to be highly non-gaussian, we use the Wilcoxon-Signed Rank non-parametric test [197].

5.6.2 Training and Testing Procedure

In the following experiments, we use the exact same training and testing procedure as in our preliminary work [99]. This allows the results to be directly comparable. In detail, we train all models using RAdam [198] on 5 s chunks obtained from training signals with 50% overlap. The last chunk is discarded if shorter. Hyperparameters such as batch size, learning rate and dropout rate are tuned for each network, dataset and training objective (speaker counting or VAD+OSD) on the development set. Inference is performed by using a sliding window approach is used where the logits of overlapping blocks are averaged to obtain the final estimate. Popular speech processing toolkits such as Pyannote [147] use this approach. In this work we use a sliding window of 3 s with 50% overlap.

In our preliminary work [99], we found that using training targets obtained via Forced-Alignment (FA) brings considerable improvement even when manual annotation is used as the ground truth in the testing phase. We also studied the efficacy of FA as an automatic labeling procedure for speech segmentation applications using synthetic data and we found that, when close-talk worn microphones are employed, it can be considered reliable even in overlapped speech regions and challenging SNR conditions. Thus, we employ FA labels to train OSDC models on both AMI and CHiME-6. In detail, we use the Kaldi [95] recipes for AMI and CHiME-6 and get the segmentation from the *tri3* GMM-HMM speech recognition model.

The results on the test set are evaluated using the official annotation, which is manual in the case of AMI and FA-based in the case of CHiME-6. In fact, the FA-based annotation of the CHiME-6 development and evaluation sets was obtained with similar FA procedure as used here.

Moreover, to further improve performance on real-world data and counteract class imbalance, we resort, in our experiments, to the data-augmentation strategy described by [99], where it was shown to bring significant improvements. This data-augmentation technique, which is itself an extension of the one proposed by [170], consists of on-the-fly creation, at training time, of new concurrent speaker examples by overlapping 2, 3, and 4 random single-speaker chunks from the original dataset in order to re-balance the classes. To further increase the training material, a random gain factor sampled from $\mathcal{N}(\mu = -16.7, \sigma = 4)$ in dB scale is applied to each chunk independently. In this way, we augmented the original AMI data by a factor of 70% and CHiME-6 data by 40 %. This augmentation factor is tuned for each dataset using the development set. In parallel, to improve generalization, we also use SpecAugment [199] on both single-channel and spatial features separately.

5.6.3 Choice of Transformer Hyperparameters and Single-Channel Features

In Table 5.2, we show the hyperparameter space explored for the proposed Transformer-based architecture. We varied number of future and past frames (C) and subsampling factor (S) used in cat-pool operation as well as size of hidden representation (H), number of attention heads, size of feed-forward neural network hidden layer (FFN size) and number of transformer encoder blocks (R). The hyperparameters were tuned on the development set of AMI, for fair comparison with [168] who also optimized his LSTM model on AMI. The models were trained to perform VAD+OSD according to the framework introduced in Section 5.2. The best combination was selected using two criteria: overall VAD+OSD performance and inference-time computational footprint, to give an overview of how much demanding the model is when used in practical applications. In fact, if the OSDC model has a modest computational burden, using it at the very first stage of a speech processing pipeline has the advantage of lowering the computational requirements of the whole pipeline, as subsequent processing can be applied only when needed. Moreover, models with modest computational requirements allow for deployment on mobile and edge-computing devices.

Table 5.2: Hyperparameter space explored for the Transformer-based architecture. The best combination of hyperparameters is highlighted in bold.

Hyperparameter	C	S	H	heads	FFN size	R
Values	(**7**, 5)	(**10**, 5)	(256, **384**)	(**4**, 8, 16)	(**1024**, 2048)	(2, **4**, 8)

In Table 5.3, we show the VAD and OSD performance on the AMI development set, as well as the total number of floating point operations (FLOP) and total memory consumption (Mem) with the best combination of hyperparameters (Best) and when changing the value of one hyperparameter at a time. FLOP and Mem are computed with a 300-frame (3 s) dummy 80-dimensional feature sequence (matching the 80 log-Mel features used in the following experiments), generated from a uniform distribution. These computational footprint figures are estimated using the built-in profiler in the Pytorch toolkit and the Performance Application Programming Interface [200]. Several observations can be made. First, the choice of hyperparameters does not affect the VAD performance, which is arguably a simpler task than OSD and is more easily tackled by the network. Second, doubling the number of Transformer Encoder blocks only marginally improves performance at the cost of a significant increase of the computational footprint. Third, increasing time resolution by halving the sub-sampling rate also significantly increases the computational requirements without bringing significant benefits, meaning that a resolution in the order of 100 ms is enough in the application scenario considered here.

Table 5.3: VAD and OSD AP (%) and computational footprint of the Transformer-based architecture on the AMI development set for different architecture hyperparameter values.

Model Parameters	FLOP [10^6]	Mem [10^6]	AP	
			VAD	OSD
Best	85.6	3.3	**98.5**	57.4
S = 5	166.8	6.9	**98.5**	57.5
R = 8	161.0	6.2	**98.5**	**57.8**
heads = 8	85.4	3.6	**98.5**	56.9
FFN size = 2048	153.1	5.1	**98.5**	**57.6**

In Table 5.4, we report the results achieved by the proposed Transformer-based architecture on the AMI development set for different choices of single-channel input features. In the past, [168] and [164] explored different single-channel features for the LSTM and CountNet architectures: [168] used gammatone filterbanks, log-Mel and other features such as kurtosis and spectral flatness, while [164] explored magnitude STFT spectra, log spectra and 40 Mel-scale filterbanks. In both studies, the features were extracted with a 25 ms window and 10 ms hop-size. Hereafter, we consider magnitude spectra computed over 32 ms and 64 ms windows (512 and 1024 samples respectively), 40 and 80 log-Mel, 40 and 80 gammatone filterbanks, and 20 and 40 MFCCs instead. All these features were computed with a 10 ms hop-size. Regarding MFCCs, we used 20 and 40 Mel bands, respectively. A window of 25 ms was used for log-Mel, gammatone and MFCCs. We can see that OSD and to a lesser extent VAD performance correlate with frequency resolution. In fact, especially for OSD, the use of compact features such as MFCCs, 40 log-Mel or 40 gammatone filterbanks leads to a loss in performance. These results partially agree with the findings of [168], who found 64 gammatone filterbanks to be superior to 40 log-Mel features for OSD.

Table 5.4: VAD and OSD AP (%) achieved by the Transformer-based architecture on the AMI development set with different choices of single-channel features.

AP	MagSpec		Log-Mel		Gammatone		MFCC	
	512	1024	40	80	40	80	20	40
VAD	**98.5**	**98.5**	98.4	**98.5**	98.4	**98.5**	98.3	98.4
OSD	**61.1**	**61.0**	58.2	**61.0**	58.0	**59.8**	56.8	58.4

Because no statistical difference was found between 80 gammatones and 80 log-Mel and higher-resolution features (e.g., 64 ms magnitude spectra) did not result in higher performance, we ultimately decided to use 80 log-Mel features in the following.

5.6.4 Computational Footprint Comparison Across Architectures

In Figure 5.6 we report the total number of floating point operations (FLOP), the total memory usage and the inference time in clock cycles for the four considered network architectures as a function of the input signal duration from 1 s to 100 s. Inference time is computed over batches of 64 examples in order to get reliable estimates. As we are interested in comparing only the architectures, we use the same single-channel features for all architectures, namely 80 log-Mel features with 25 ms window and 10 ms hop-size. An Intel i9-10920X CPU is employed to perform the comparison.

As expected, regarding inference speed, the RNN-based architectures (LSTM and CRNN) are slower than the TCN and the Transformer, which do not employ recurrence. A similar trend is observable in the FLOP plot, with the difference that the CRNN has a much higher FLOP count than the other architectures due to the use of 2-D convolutions, despite the fact that it is slightly faster than the LSTM architecture as it employs pooling operations and the CNN part is parallelizable. The use of 2-D convolutions also increases the CRNN memory footprint with respect to the other architectures. The small number of parameters employed in the TCN leads to similar memory footprint as the LSTM architecture.

Overall, the proposed Transformer architecture is the most efficient according to the three criteria despite having the second largest number of parameters after the LSTM. Due to the cat-pool operation, the total memory usage is kept contained and grows almost linearly until a duration of 100 s. In practice, due to the fact that OSDC typically requires a context of a few seconds only,

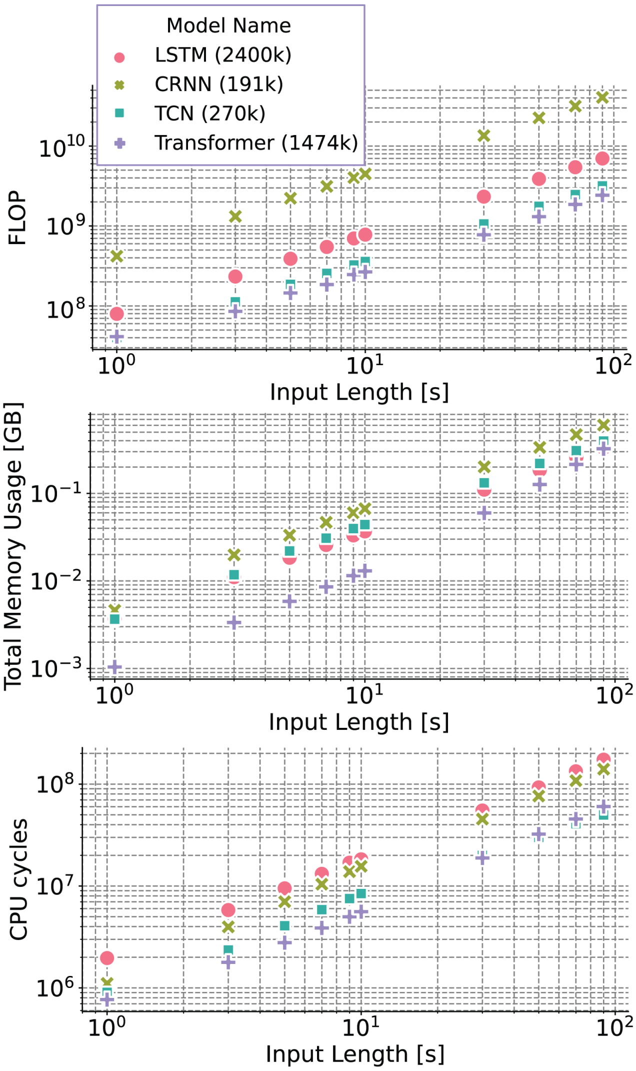


Figure 5.6: Inference-time computational footprint for the four considered neural network architectures as a function of the input signal duration. Top: number of floating point operations (FLOP). Middle: Total memory usage in GB. Bottom: number of CPU clock cycles. The numbers in parentheses in the legend indicate the number of model parameters. The two axes are in log-scale.

inference is never performed directly over such long signals. In fact, a sliding window approach, as explained in Section 5.6.2 is employed. More generally, all architectures, including previously proposed LSTM and CRNN ones, attain overall computational resources figures which are suitable for edge devices deployment. For example, the FLOP count for one second of audio is comparable to the one reported by [201] for keyword spotting in smartphone devices.

An important take from these results is also that the number of parameters, which is widely used as a gauge for model computational burden, does not correlate well with the latter and can be deceptive when comparing very different architectures.

5.6.5 Single-Channel Experimental Results

We now evaluate the performance achieved by the four architectures on the AMI and CHiME-6 distant speech datasets. For the sake of comparison with [168] and [164], we use single-channel features only, namely 80 log-Mel features with 25 ms window and 10 ms hop-size.

Each architecture is trained and evaluated according to two different tasks: VAD+OSD and speaker counting. Indeed, we are interested in assessing the feasibility of VAD+OSD and speaker counting on real-world data. Speaker counting, as already said, has the advantage of providing more information to downstream tasks, but it is plagued by extreme class imbalance. VAD+OSD, by contrast, does not provide any clue about concurrent speakers, but exhibits a less extreme class imbalance.

Concerning AMI, to allow direct comparison with previous works [99, 168], data from all microphone channels is used during training while testing is performed on the first microphone of array 1. Regarding CHiME-6, training is also performed using all microphone channels from all array devices but, when evaluating, we consider for each array the first channel and then average the outputs of single-channel systems across all arrays because of the multi-room environment of CHiME-6.[4]

In Table 5.5, we report the VAD and OSD results obtained when training the models with a VAD+OSD objective. It can be seen that the AP figures on both datasets are considerably higher for VAD than for OSD. This is expected since OSD is inherently a more challenging task than VAD. As also expected, the performance is better on AMI than CHiME-6, as CHiME-6 is arguably a much more challenging dataset, having lower SNR due to the more unconstrained setting. The proposed Transformer architecture performs on-par or better than the other architectures, with the TCN architecture closely following. LSTM and CRNN perform significantly worse, despite the addition of normalization layers which were not present in the respective original works of [168] and [164].[5]

[4] The single-channel evaluation protocol for CHiME-6 differs from the multichannel protocol adopted by [99], who averaged the outputs of single-channel systems across all 24 microphones instead.

[5] These normalization layers do improve performance, as can be seen by comparison with

Table 5.5: VAD and OSD AP (%) achieved by the four considered neural network architectures on the AMI and CHiME-6 evaluation sets using single-channel features and VAD+OSD as a training objective.

VAD+OSD Model	VAD		OSD	
	AMI	CHiME-6	AMI	CHiME-6
LSTM	95.4	93.4	34.3	28.7
CRNN	96.7	93.8	38.9	33.2
TCN	**98.5**	**94.3**	54.2	49.0
Transformer	**98.5**	**94.3**	**57.8**	**49.9**

Similarly, Tables 5.6 and 5.7 report the speaker counting results achieved on the evaluation sets of AMI and CHiME-6, respectively, when training the models with a counting objective. The fact that the AP for the 0-spk class is remarkably lower on AMI is a rather unexpected result, as it features a much higher SNR than CHiME-6 overall. This could be explained by class imbalance since, as reported in Table 5.1, the proportion of 0-spk in AMI is significantly lower than in CHiME-6. The proposed Transformer architecture achieves the best figures overall on both datasets. In general, compared to the 0-spk and 1-spk classes, the AP degrades considerably for the 2-spk class and even more so for the 3-spk and 4-spk classes. This suggests that the data-augmentation strategy, is only able to partially compensate for the extreme imbalance of 3-spk and 4-spk classes. Therefore, it can be said that speaker counting is still far from being reliable on real-world data.

Table 5.6: Speaker counting AP (%) achieved by the four considered neural network architectures on the AMI evaluation set using single-channel features and counting as a training objective.

Counting Model	0-spk	1-spk	2-spk	3-spk	4-spk
LSTM	47.0	82.4	24.7	6.4	**0.02**
CRNN	49.8	84.2	34.8	9.2	**0.03**
TCN	**50.7**	86.1	40.4	11.3	**0.03**
Transformer	**50.9**	**87.2**	**41.8**	**11.2**	**0.03**

In Table 5.8 we compare the performance of Transformer models trained to perform either VAD, OSD, VAD+OSD or counting for the VAD and OSD tasks. For each dataset, we report the evaluation set performance and, in parentheses, the development set performance. Regarding VAD, the choice of the training objective has little impact on performance on all datasets. Regarding OSD, in-

the results reported in our preliminary work [99] which did not include them.

Table 5.7: Speaker counting AP (%) achieved by the four considered neural network architectures on the CHiME-6 evaluation set using single-channel features and counting as a training objective.

Counting Model	0-spk	1-spk	2-spk	3-spk	4-spk
LSTM	79.1	69.7	20.5	6.1	**0.002**
CRNN	86.2	73.8	25.4	8.5	**0.003**
TCN	**88.3**	**77.3**	30.0	**12.3**	**0.003**
Transformer	**88.2**	**77.3**	**30.6**	**12.5**	**0.003**

terestingly, the model trained to perform speaker counting, which is inherently a more difficult task, leads to better OSD performance than the model trained directly with a VAD+OSD or OSD objective on the AMI development and evaluation sets and on the CHiME-6 evaluation set. This is especially evident on AMI, where a larger gap between the two models is observed. So, while speaker counting performs poorly on real-world data, it can be convenient to use models trained to perform speaker counting to perform VAD and OSD instead. This may be explained by the fact that speaker count labels provide the model with more information during training than mere OSD labels.

Table 5.8: VAD and OSD AP (%) achieved by the Transformer-based architecture on the AMI and CHiME-6 development and evaluation sets when using single-channel features and either VAD, OSD, VAD+OSD or counting as the training objective. The values obtained on the development sets are in parentheses.

Method	VAD		OSD	
	AMI	CHiME-6	AMI	CHiME-6
Transformer-VAD	**98.5** (**98.6**)	**94.3** (**93.2**)	n.a.	n.a.
Transformer-OSD	n.a.	n.a.	57.8 (61.0)	50.2 (55.4)
Transformer-VAD+OSD	**98.5** (**98.6**)	**94.3** (93.1)	57.8 (61.0)	49.9 (55.1)
Transformer-Counting	**98.5** (98.5)	**94.3** (**93.2**)	**59.1** (**64.3**)	**50.8** (**55.8**)

5.6.6 Multichannel Experimental Results

In the following, we select the best model found in Section 5.6.5, namely the proposed Transformer model trained with a speaker counting objective, and we show how its performance can be improved by employing spatial features along with single-channel features. To do so, we evaluate the IPD, CSIPD and neural network-based spatial features and the early and late fusion schemes described in Section 5.4 using AMI, CHiME-6 and the proposed synthetic dataset.

In order to allow direct comparison with single-channel results, we adopt the same training strategy as above. Data augmentation is extended to the multichannel scenario by overlapping multichannel audio chunks and being careful, when mixing, in maintaining the array topology (i.e., the first channel is always mixed with the first channel). Training is performed by considering each array separately and using the same FA-based targets as above. Testing is performed, on AMI and CHiME-6, by averaging the predictions made independently for each array across all arrays (i.e., 2 devices for AMI and 6 for CHiME-6).

The IPD and CSIPD features are computed with an STFT window length of 50 ms and the same 10 ms hop-size as single-channel log-Mel features. The corresponding feature vectors, for each microphone pair, are thus of size 801 and 1602, respectively.

Neural network based localization features are extracted using the same Transformer-based architecture as for OSDC, but with $R = 2$ and the modifications outlined in Section 5.4.2. The network takes CSIPD features relative to most distant microphone pairs with the same STFT window length and hop-size as above, and it outputs $D = 181$ discrete DoAs. It is trained on matched synthetic datasets. More specifically, concerning AMI, we use our synthetic dataset by simulating a circular array instead of the linear one and compute CSIPDs over the 4 pairs obtained by taking opposing microphones in the circular array.

Regarding CHiME-6, we perform training on the Kinect-WSJ2Mix dataset [202] which involves simulated Kinect devices and real CHiME-6 noise and we use CSIPD features between the 3 microphone pairs with largest distance, as explained in Section 5.4.1. Because Kinect-WSJ2Mix involves at most 2 overlapping speakers while in CHiME-6 up to 4 concurrent speakers can be present, we extend the original data by creating on-the-fly mixtures of up to 4 overlapped speakers and use this newly generated data to train the localization network.

Regarding the experiments performed on the synthetic dataset, we also use CSIPD features between the 3 microphone pairs with largest distance as inputs. Contrary to the AMI and CHiME-6 real-world datasets, in which the localization network is trained on a separate dataset, here we use the same synthetic data for both the OSDC and the localization network.

In addition, to avoid possible domain mismatch between the simulated training dataset for the localization network and the test dataset for the OSDC network, we fine-tune the localization network with the OSDC model by joint optimization with respect to the speaker counting task on the OSDC training dataset. This fine-tuning step is critical to achieve good performance when applying the OSDC network to real-world datasets: for example, on CHiME-6 without fine-tuning the resulting AP is in the order of 50% only. We summarize the datasets used for training the neural localization network, fine-tuning and testing with the back-end OSDC system in Table 5.9.

Table 5.9: Datasets used for the neural localization network experiments: training (*train*), fine-tuning (*adapt*) with OSDC back-end and testing (*test*) dataset splits. The total number of hours for each dataset is reported in parenthesis.

Datasets		
Localization Network	OSDC Network	
train	adapt	test
Synthetic (23h)	AMI (81h)	AMI (9h)
Reverberated WSJ-2mix (47h)	CHiME-6 (40.3h)	CHiME-6 (5.2h)
Synthetic (23h)	Synthetic (23h)	Synthetic (4.6h)

In Tables 5.10 and 5.11, we report the performance achieved for the VAD and OSD tasks, respectively, with different spatial features, fusion schemes, and numbers of microphone pairs. Microphone pairs are selected as described in Section 5.4.1, by considering, as the upper bound (*all*), only pairs which add significant spatial diversity, i.e., from 1 to 4 pairs formed by opposing microphones in AMI and from 1 to 3 pairs in CHiME-6 and the synthetic dataset. We also include in the comparison of a single-channel ensemble system with no spatial features, where ensembling is done by averaging the OSDC network outputs over all microphones in the array and a single-channel system trained on beamformed audio using BeamformIt [70].

Table 5.10: VAD AP (%) achieved on the AMI, CHiME-6 and synthetic evaluation sets by the Transformer-based architecture trained with a speaker counting objective for different spatial features, fusion schemes, and numbers of microphone pairs (1, 2 or all), as compared to single-channel features only (*None, 1 ch.*), an ensemble of single-channel systems (*None, all ch.*) and a single-channel system + BeamformIt (*None, enh*).

Dataset	Fusion	IPD			CSIPD			Neural	None		
		1	2	all	1	2	all	all	1 ch.	all ch.	enh
AMI	early	98.6	**98.7**	**98.7**	98.6	**98.7**	**98.7**	**98.7**	98.5	98.6	98.5
	late	98.6	**98.7**	**98.7**	98.6	**98.7**	**98.7**	**98.7**			
CHiME-6	early	94.7	94.8	94.8	94.7	94.9	95.1	**95.4**	94.3	94.5	94.3
	late	94.8	95.4	95.4	94.9	95.4	95.4	**95.5**			
Synth	early	96.3	96.8	97.2	96.1	96.4	96.8	**97.5**	96.4	96.6	96.4
	late	96.5	97.2	97.4	96.3	97.1	97.4	**97.5**			

For what concerns VAD performance in Table 5.10, it can be seen that neural network-based localization features result in on-par or higher performance than the other spatial features, and they outperform single-channel systems by a significant margin on CHiME-6 and the synthetic dataset. Regarding AMI, the AP saturates for most models due to the fact that, as noted previously in Section 5.6.5, silence is under-represented in the material. An interesting trend which appears on CHiME-6 and synthetic data is that the performance of signal-based spatial features improves when increasing the number of microphone pairs and by using late fusion. Especially on models with late fusion, using more microphones considerably boosts the performance for IPD and CSIPD features. Instead, a smaller improvement is noticeable when early fusion is employed, due to the fact that the size of CSIPD and IPD features grows linearly with the number of pairs but the bottleneck convolutional layer applied in early fusion maps them to a fixed-size representation (384 neurons, as reported in Table 5.2). Thus some information is inevitably lost in early fusion. On top of that, in late fusion spatial features are available at multiple stages of the architecture.

Table 5.11: OSD AP (%) achieved on the AMI, CHiME-6 and synthetic evaluation sets by the Transformer-based architecture trained with a speaker counting objective for different spatial features, fusion schemes, and numbers of microphone pairs (1, 2 or all), as compared to single-channel features only (*None, 1 ch.*), ensemble of single-channel systems (*None, all ch.*) and a single-channel system + BeamformIt (*None, enh*).

Dataset	Fusion	IPD			CSIPD			Neural	None		
		1	2	all	1	2	all	all	1 ch.	all ch.	enh
AMI	early	58.1	58.6	**59.4**	57.8	58.4	58.9	**59.3**	57.8	58.6	57.6
	late	58.4	59.5	**60.3**	58.1	59.6	**60.4**	59.7			
CHiME-6	early	51.4	51.5	51.6	51.3	51.4	51.5	**51.8**	50.8	51.2	50.2
	late	51.6	**52.4**	**52.4**	51.7	**52.3**	**52.2**	51.9			
Synth	early	81.8	82.3	82.7	81.6	82.0	82.4	**83.8**	82.4	83.1	82.1
	late	82.8	83.4	84.2	82.9	83.6	**84.4**	**84.3**			

Similar trends can be also observed for OSD performance in Table 5.11 regarding the number of microphone pairs and early fusion versus late fusion. Notably, neural network-based spatial features are outperformed by signal-based ones on AMI and CHiME-6 when late-fusion is used but reach on-par or top performance when early fusion is employed instead. This suggests that fine-tuning the localization network compensates for the synthetic/real domain mismatch only up to a certain point regarding OSD. It can also be observed that the performance gain achieved by late fusion with respect to early fusion appears modest for neural spatial features, while it is substantial for signal-based ones. This is explained by the fact that neural network-based features are less affected by the aforementioned “bottleneck issue” in early fusion, as they have a more compact size than signal-based ones and, moreover, are jointly fine-tuned with the OSDC system. Again, models with spatial features are able to outperform the single-channel systems and ensembles of single-channel systems. This is notable, as the ensemble is performed using all channels in the array and it comes at the cost of increasing the computational footprint linearly in the number of channels. By contrast, spatial features allow us to boost performance with a smaller increase in computational requirements. The use of beamformed audio degrades OSD performance but not VAD performance with respect to the single-channel only baseline system. This could be explained by the fact that BeamformIt tends to enhance the source with the highest energy and attenuate the rest.

In Tables 5.12 and 5.13 we report the counting performance achieved for different spatial features on AMI and CHiME-6, respectively, using two microphone pairs and late fusion. On both datasets, a similar trend can be noticed. On the one hand, neural network based localization features achieve the best figures regarding the 0-spk and 1-spk classes which are the most represented ones. This is in accordance with the VAD results in Table 5.10 where neural spatial features have in general higher scores. On the other hand, CSIPD and IPD obtain similar or higher AP values for 2 and 3 concurrent speakers. This is in accordance with the OSD results in Table 5.11. Nonetheless, while systems based on spatial features are able to substantially increase the speaker counting performance over single-channel systems, the observations made in Section 5.6.5 are still valid, and reliable speaker counting remains out of reach on real-world data.

Table 5.12: Speaker counting AP (%) achieved on the AMI evaluation set by the Transformer-based architecture trained with a speaker counting objective for different spatial features, as compared to single-channel features only (*None, 1 ch.*) or an ensemble of single-channel systems (*None, all ch.*).

Spatial Features	0-spk	1-spk	2-spk	3-spk	4-spk
IPD	52.8	88.3	**45.0**	**12.8**	**0.03**
CSIPD	52.9	88.4	**45.1**	**12.7**	**0.03**
Neural	**53.1**	**88.8**	44.9	11.8	**0.03**
None, 1 ch.	50.9	87.2	41.8	11.2	**0.03**
None, all ch.	51.3	87.9	42.4	11.5	**0.03**
None, enh	50.8	87.4	41.6	10.8	**0.03**

Table 5.13: Speaker counting AP (%) achieved on the CHiME-6 evaluation set by the Transformer-based architecture trained with a speaker counting objective for different spatial features, as compared to single-channel features only (*None, 1 ch.*) or an ensemble of single-channel systems (*None, all ch.*).

Spatial Features	0-spk	1-spk	2-spk	3-spk	4-spk
IPD	89.9	78.8	**32.6**	**12.4**	**0.003**
CSIPD	90.1	78.7	**32.5**	**12.4**	**0.002**
Neural	**90.2**	**79.0**	32.2	11.9	**0.003**
None, 1 ch.	88.2	77.3	30.6	12.5	**0.003**
None, all ch.	90.1	78.4	31.4	11.9	**0.003**
None, enh	88.1	77.4	30.3	11.8	**0.003**

Finally in Figure 5.7 we use the synthetic dataset to further explain the benefit of spatial features. Using mixtures of two speakers, we report the OSD AP values obtained by the system using single-channel features only versus the ones obtained with late fusion and CSIPD features computed using the 3 microphone pairs with largest distance. The OSD AP performance is plotted against the mean distance of the two speakers from the array and the angle between them as seen from the array. It can be seen that, for the single-channel model, performance degrades to some extent as the speaker distance increases (i.e., colors become darker from bottom to top), but it is largely independent of the angle between the speakers. By contrast, for the model employing spatial features, performance still degrades as the speaker distance increases but at the same time it clearly improves as the angle between the speakers increases (i.e., colors become lighter from left to right). In fact, the AP is significantly boosted for angles greater than 30 degrees, indicating that spatial features offer complementary information which allows the model to more effectively discriminate frames with overlapped speech.

5.7 Conclusions & Future Work

In this Chapter, we presented a brief state-of-the-art and historical overview of VAD, OSD and speaker counting. We then reported a study about VAD+OSD and speaker counting on real-world meeting scenarios recorded with distant microphone arrays. We focused on neural network based approaches and compared different architectures for the two tasks, on AMI, CHiME-6 and a purposedly developed synthetic dataset. Among the neural networks compared we introduced two novel architectures: one based on TCNs and another based on the Transformer. In parallel we explored the use of spatial features, both

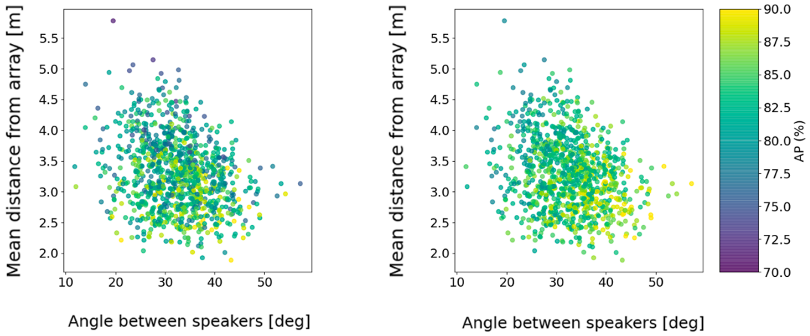


Figure 5.7: OSD AP (%) achieved on the synthetic evaluation set by the Transformer-based architecture trained with a speaker counting objective as a function of the mean distance of the speakers from the array and the angle between the speakers. Left: single-channel features only. Right: CSIPD spatial features and late fusion.

signal-based and neural-based, to aid in the VAD+OSD and speaker counting tasks when multiple microphones are available. We conducted an extensive experimental evaluation by comparing the models's computational footprint and VAD, OSD and counting performance on single-channel and multichannel distant speech data. On CHiME-6, our proposed TCN and Transformer-based architectures achieve an absolute improvement in AP of 15% and 16% over previous techniques, respectively. Overall, we found the proposed Transformer-based architecture to be the most promising as it was shown to be able to reach on-par or better results than the other architectures with a significantly lower computational footprint. In general, in comparing VAD+OSD and speaker counting tasks we found that, due to class imbalance, speaker counting performs poorly on real-world data, but, on the other hand, it is desirable to use a speaker counting objective to train a system to perform VAD+OSD as it is shown to improve OSD. Finally, concerning spatial features, we found that significant further improvements can be obtained by using a late-fusion strategy and by increasing the number of microphone pairs considered. Neural-based spatial features show a clear advantage over signal-based ones for VAD across all datasets, but no spatial feature shows a clear advantage over another for OSD or counting. Future work includes fusing estimates over multiple arrays in a way that favors arrays closer to the speakers and exploits the relative positions and orientations of the arrays whenever they are known.

Chapter 6

Leveraging Speech Separation for Low-Latency Speaker Diarization

Context

This work was presented at SLT 2022 [203] and is an equal contribution with Giovanni Morrone. It was done within the AGEVOLA project in collaboration with Desh Raj from John Hopkins University, Enrico Zovato from PerVoice s.r.l. Alessio Brutti from Fondazione Bruno Kessler and Luca Serafini from UNIVPM. The following section about the history and current state of the art diarization is part of a review article [204] done also within the AGEVOLA project.

6.1 A Brief History of Speaker Diarization

Speaker diarization (or diarisation), also often referred to as simply diarization, aims at segmenting an audio recording into temporal segments denoting the boundaries of each speaker's utterances. It addresses the problem of "who spoke when?", without a-priori knowledge of the speakers' identities and is an essential front-end task for many applications, such as meeting transcription, live captioning, speaker-based indexing, and telephone conversation analysis to name a few.

The Early Days

The first works on speaker diarization can be traced back to the 1990s [205–209] with these early works focusing on applications such as radio broadcast news and communications. The focus of these early works was ASR speaker adaptation and indeed some relied on features derived from the ASR outputs directly [207, 209] (e.g. two pass decoding). The use of speech separation for performing diarization was proposed by [206] again mainly for ASR applications.

Importantly, some of these early works [205, 208], laid the foundations for the clustering-based diarization paradigm (Figure 6.1) which would be the de-facto standard approach for decades to come. They realized that diarization was best addressed at the time as a clustering problem. The input audio stream was first segmented using VAD, and then on each segment, speaker discriminative features are extracted with a sliding window approach (chunking block in Figure 6.1). These speaker discriminative features are then clustered together in order to assign each of the original chunks to each speaker. The number of clusters is also used to detect the total number of speakers in the recording. The clustering step was largely based on agglomerative hierarchical clustering (AHC) and common measures or criterion to define similarity (or distances) between the speaker features were bayesian information criterion (BIC) [208] and generalized likelihood ratio (GLR) [205]. In these early days the features used were mostly “hand-crafted": Mel-spaced frequency cepstrum coefficients (MFCC), perceptual linear predictive (PLP) [210], linear predictive coding (LPC) [211] features were a common choice.

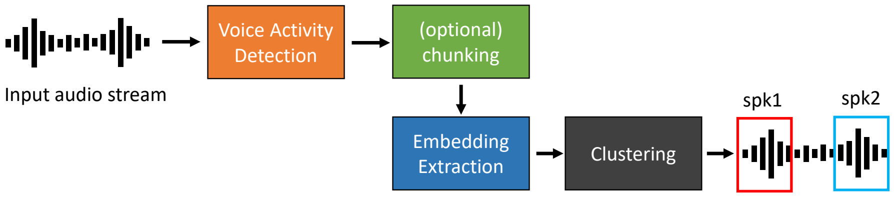


Figure 6.1: General block scheme for a clustering-based diarization system.

The 2000s and Beyond

During the first decade of the new millennium researchers understood the need to move from fully hand-crafted to principled data-driven methods to obtain more robust and higher-level speaker-id discriminative features. A significant advancement in this sense was done by Reynolds et al. [212]. This work introduced the speaker-independent gaussian mixture model (GMM) universal background model (GMM-UBM altogether) for speaker verification. In this new paradigm, each vector of features is derived in a data-driven fashion from a GMM, a probabilistic generative model that represents the data with a weighted sum of a finite number of multi-dimensional Gaussian components. The main idea is that this GMM-UBM model could be trained on a large amount of data with a large speaker identity variability via a maximum a posteriori (MAP) criterion. Then, for diarization applications, after VAD some parameters Such GMM-UBM paradigm remained the mainstream approach to diarization since the invention of DNN-based speaker discriminative features.

Other key works of this decade built upon the GMM-UBM paradigm trying to address its shortcomings [213–215]. The research focused on boosting speaker discriminative features robustness against intra-speaker variability (e.g., due to changes in intonation, background noise, etc.) and, at the same time, inter-speaker discriminability. This latter to allow for better differentiating distinct speakers. For example, joint factored analysis (JFA) [213] and Eigenvoice priors [214] try to tackle these issues by exploiting lower dimensional factorizations of the GMM supervectors, to use as a more robust speaker-dependent features with lower intra-speaker variability. JFA in particular assumes that the supervector covariance matrix can be decomposed into a channel space and speaker space, with the channel space responsible for the intra-speaker variations. It then introduces the concept of speaker and channel-dependent supervectors and uses these two to obtain disentangled speaker representations by factoring out the channel-dependent component.

While effective, Dehak et al. [215] found that this decomposition is far from perfect and speaker-related information tend to leak into channels factors. They propose instead to define just only total variability matrix which models jointly the channel and speaker factor simultaneously and not two independent channel and speaker spaces as in JFA. Speaker id features then can be obtained via a projection of this total variability space. This can be done for each utterance through Baum-Welch statistics. They call this projection vector *i-vector* and it found to be a the most effective pre-deep learning speaker discriminative feature. Channels effect are compensated via linear discriminant analysis (LDA) [215] or via [216] probabilistic LDA (PLDA). Such use of PLDA and i-vectors has been a popular technique for speaker verification and diarization till the advent of deep learning based approaches.

The Deep Learning Era

Starting from 2014, the studies and refinements in the deep learning area, together with the increasing availability of annotated transcriptions and data, made it possible to exploit DNNs in place of GMMs for obtaining speaker-discriminative features. One of the first works in this direction is the one of Variani et al. [217], in which DNN-based features (so-called d-vectors) were shown to be able to outperform i-vectors, the state-of-the-art approach of the time, especially in noisy conditions. As it happens with the GMM-UBM, in [217] the DNN was trained on large corpora with a vast number of speakers and different acoustic conditions to try to classify the correct speaker (multi-class classification problem) among all the ones in the training set. In inference then the output of the last hidden layer was used as a speaker-discriminative feature for speaker verification or diarization.

A very popular and effective follow-up work, is the invention of the x-vector extractor [172] which employs a time-delay neural network (TDNN) and a statistical pooling layer to obtain a low-dimensional speaker-id representation. This trend of designing better DNN architectures to improve speaker-id discriminative features is continuing today [218]. Some recent improvements include the use of ResNet-based designs [219], TitaNet [220], ECAPA-TDNN [221] and the use of self-supervised learning pre-trained models such as WavLM [15]. Other works [222–224] instead have focused on the loss function and training strategy to use to train such a DNN speaker-id feature extractor. For example, [222, 224] proposed to use metric learning approaches, whereas [223] the use of angular-softmax loss to improve performance. This is also a very active research direction. In all these works, clustering continued to be, as said, the main approach with DNN-based speaker-id representation and was used often in conjunction with PLDA to reduce dimensionality and intra-speaker variability before clustering.

Indeed many advances also regarded other components of the diarization pipeline, such as the clustering step or the post-processing step. For example, Park et. al. [225] showed that by leveraging spectral clustering is possible to improve over the at the time state-of-the-art PLDA followed by AHC approach. Another notable work, this time regarding post-processing, is variational Bayesian (VB) resegmentation, initially proposed for an i-vector-based system [226] and later adapted to an x-vector-based system [227]. This latter approach called VBx has been proven extremely effective on a wide number of datasets [228] and challenges [229]. Among the post-processing works it is worth mentioning ensembling or fusion methods that allow combining the output of multiple heterogeneous diarization systems in order to improve the performance. Such works include diarization output voting error reduction (DOVER) [230] and DOVER-Lap, a recent improvement over DOVER that allows handling also overlapped speech regions [231, 232].

A number of recent works also explored how to improve clustering-based methods with deep learning based techniques in order to let them deal also with overlapped speech. For example, Bullock et al. [170] proposed to use an overlap detector to mask the speaker posterior matrix in the VBx method. Raj et al. [233] instead devised a way to handle speaker-id discriminative features from overlapped speech regions during the clustering step.

Due to the greater availability of annotated data and possibilities opened up by deep learning recently a new line of research arose, involving other alternative approaches to diarization that try a tighter integration with DNNs. Among such works are region-proposal networks diarization (RPND) [234], unbounded interleaved-state recurrent neural network (UIS-RNN) [235], discriminative neural clustering (DNC) [236], deep learning SSGD-based approaches

[203, 237], target-speaker VAD (TS-VAD) [238]. Some of these approaches and in particular SSGD and TS-VAD have proven to be particularly effective: for example, the system winning the recent third DIHARD Challenge [239], was based on a combination of SSGD and TS-VAD based approaches [240]. UIS-RNN [235] and DNC [236] consist of supervised neural models based on speaker embeddings. The former is clustering-free, whereas the latter relies on neural-based clustering. They are both not overlap-aware, but they are able to manage a variable number of speakers.

Building on the invention of PIT for DNN-based speech separation in 2019 [241], Fujita et al. developed the first fully end-to-end DNN-based system [242] which was later improved using self-attention [243]. This again sparkled another research direction on systems based on end-to-end neural diarization (EEND) which is very active nowadays. Horiguchi et al. [244] proposed to extend EEND with an encoder-decoder-based attractor architecture (EEND-EDA) able to handle a flexible number of output speakers thanks to an autoregressive decoder.

Kinoshita et al. proposed several improvements to the initial EEND approach by integrating speaker-id embeddings extraction so that the strengths of EEND and clustering-based approaches can be combined in a framework called EEND-vector clustering (EEND-VC) [245–248].

Other recent works focused on streaming processing: [249] proposed to extend EEND with a speaker-tracing buffer to solve the permutation ambiguity caused by PIT when the model inference is performed via sliding windows. EEND-EDA streaming versions have been also recently proposed [250, 251]. Also focusing on online processing, [252], combined the use of EEND with an x-vector extractor and online clustering, where the EEND model is used to gate the representation before the x-vector statistical pooling layer, to extract per-speaker embeddings even in overlap regions. Recent results in the most popular diarization challenges indicate that EEND-based systems are increasingly competitive [240, 253, 254] and nowadays surpass clustering-based methods.

Finally, other end-to-end approaches, that do not rely on PIT, such as DIVE [255] or joint speaker diarization and ASR systems based on a recurrent neural network transducer (RNN-T) [256], have been also recently proposed.

6.2 Low-latency Speech Separation Guided Diarization with Leakage Removal

A classical speech-separation guided diarization pipeline [237] is composed of two main modules: a speech separation algorithm and a VAD.

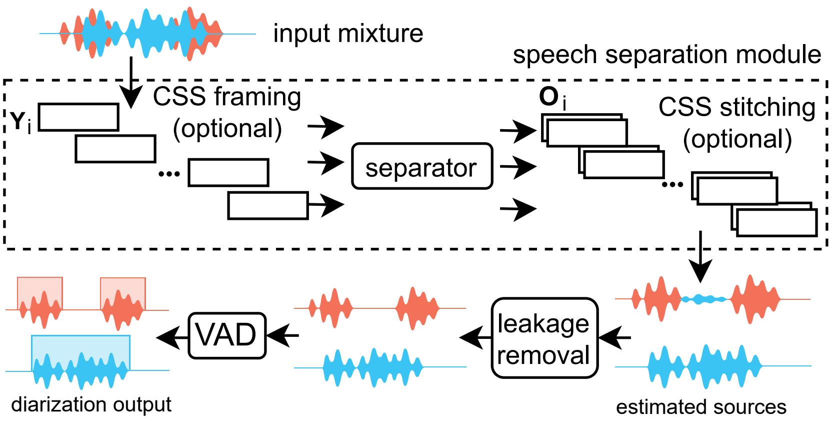


Figure 6.2: General diagram for the SSGD method.

In our recent work [203] we propose the addition of a third module, a leakage-removal post-processing step whose goal is to reduce the false alarm due to leaked speech in single speaker segments. Our newly proposed SSGD is shown in Fig. 6.2. The input of the system is a single-channel mixed audio stream, denoted as $\mathbf{Y} \in \mathbb{R}^{1 \times T}$, where T is the number of audio samples. In Fig.6.2 we also consider the possibility that the speech separation is done independently for each chunk, and the result is aggregated using continuous speech separation (CSS) [257]. This is different from [258, 259] where instead a classical diarization system is used to resolve the permutation ambiguity between neighboring CSS chunks.

6.2.1 Speech Separation Module

We consider in our experiments SSGD based on causal separation models (i.e., Conv-TasNet [176] and dual-path recurrent neural network (DPRNN) [54]). Since the majority of diarization approaches only work offline, we also experiment with non-causal separation models (as used in [237]) to carry out a more comprehensive comparison with clustering-based and EEND-based state-of-the-art systems. Additionally, we analyze the application of CSS with non-causal speech separation (SSep) models. In such configuration, the latency of these models is tied to the CSS window size and thus can be used online. CSS is not applied to causal SSep models since they are already capable to process the input in a streaming fashion with low latency.

Briefly, CSS consists of three stages as shown in Fig. 6.2: framing, separation and stitching. In the framing stage, a windowing operation splits $\mathbf{Y}$ into I

overlapped frames $\mathbf{Y}_i \in \mathbb{R}^{1\times W}, i = 1, \dots, I$, with $I = \lceil \frac{T}{H} \rceil$, where W and H are the window and hop sizes, respectively. Then, separation is performed independently on each frame $\mathbf{Y}_i$, generating separated output frames $\mathbf{O}_i \in \mathbb{R}^{C\times W}$, where C is the number of output channels. In this work, C is fixed to 2, meaning that we assume that the maximum number of speakers in any frame is 2. This is a common assumption made for CSS systems, and is also valid in general for telephone conversations (which is the focus of this work). To solve the permutation ambiguity between consecutive frame outputs, the stitching module aligns channels of two separation outputs $\mathbf{O}_i$ and $\mathbf{O}_{i+1}$ according to the cross-correlation computed on the overlapped part of consecutive frames. The final output stream $\mathbf{X} \in \mathbb{R}^{C\times T}$ is generated by an overlap-add operation with an Hanning window.

6.2.2 Leakage Removal Post-Processing

In the presence of long input recordings, even state-of-the-art separation models are prone to channel leakage when only one speaker is active (e.g., see *estimated sources* in Fig. 6.2). As a result, the “leaked” segments are detected as speech by the following VAD module, leading to a higher false alarm error in the final diarization output. To alleviate this problem, we propose a post-processing algorithm to reduce false alarms without significantly affecting missed speech, speaker confusion errors, and separation quality. It does not introduce additional latency and its computational overhead is negligible.

Given an input mixture $\mathbf{Y}$ and two estimated sources $\mathbf{X}^1$ and $\mathbf{X}^2$, we split each signal into disjoint segments $\mathbf{Y}_\ell$, $\mathbf{X}^1_\ell$, $\mathbf{X}^2_\ell$ of length L. For each segment, we compute the SI-SDR [260] s^1_ℓ, s^2_ℓ between segments of every source $\mathbf{X}^1_\ell$, $\mathbf{X}^2_\ell$ with the associated segment $\mathbf{Y}_\ell$ of input mixture. If both s^1_ℓ, s^2_ℓ are above a threshold $t_{\ell r}$, a segment with leakage is detected. Leakage is removed by filling with zeros the segment with lower SI-SDR. This process results in new estimated sources $\tilde{\mathbf{X}}_\ell$, which are passed as input to the VAD module. The leakage removal algorithm is summarized in the pseudo-code below.

Algorithm 1 Leakage Removal

```
Input: $\mathbf{Y}$, $\mathbf{X}^1$, $\mathbf{X}^2$, $T$, $L$, $t_{\ell r}$
Output: $\tilde{\mathbf{X}}_\ell{}^1$, $\tilde{\mathbf{X}}_\ell{}^2$
$\tilde{\mathbf{X}}_\ell{}^1 \leftarrow \mathbf{X}^1$; $\tilde{\mathbf{X}}_\ell{}^2 \leftarrow \mathbf{X}^2$
for $i \leftarrow 0$ to $T$ by $L$ do
    $s_\ell^1 \leftarrow$ SI-SDR($\mathbf{Y}[i{:}i{+}L]$, $\mathbf{X}^1[i{:}i{+}L]$)
    $s_\ell^2 \leftarrow$ SI-SDR($\mathbf{Y}[i{:}i{+}L]$, $\mathbf{X}^2[i{:}i{+}L]$)
    if $s_\ell^1 > t_{\ell r}$ and $s_\ell^2 > t_{\ell r}$ then
        if $s_\ell^1 > s_\ell^2$ then
            $\tilde{\mathbf{X}}_\ell{}^2[i{:}i{+}L] \leftarrow 0$
        else
            $\tilde{\mathbf{X}}_\ell{}^1[i{:}i{+}L] \leftarrow 0$
```

6.2.3 Voice Activity Detection (VAD)

The VAD module is used to extract active speech segments from the post-processed estimated sources and generate the diarization output. It is applied on each estimated source $\tilde{\mathbf{X}}_\ell$ independently but future work could also consider a multi-source VAD. We experiment with two different VAD models: an energy-based VAD [219], and a neural model which employs a temporal convolutional network (TCN), as proposed in [99, 128] and already outlined in Chapter 5.

6.3 Experimental Setup

6.3.1 Datasets

Since the focus of our work is on the conversational telephone speech (CTS) scenario, we use the *Fisher Corpus Part 1* [28] for both training and test purposes. Fisher consists of 5850 telephone conversations in English, sampled at 8 kHz, between two participants. It provides a separated signal for each of the two speakers. This allows training a separation model directly on this dataset and computing source separation metrics such as the SI-SDR improvement (SI-SDRi). Training, validation and test sets are created by drawing 5728, 61, and 61 conversations, respectively, with no overlap between speakers identities. The amount of overlapped speech is around 14% of the total speech duration.

In addition, we generate a simulated fully-overlapped version of Fisher for the purpose of training the SSep models. This portion is derived from the training set and amounts to 30000 mixtures for a total of 44 hours.

We also test the proposed methods on the portion of the 2000 NIST SRE [27, 261] denoted as *CALLHOME*, consisting of real-world multilingual telephone conversations. Following the recipe in [179], we use the 2-speaker subset of CALLHOME and the adaptation/test split that allows to compare with most end-to-end diarization methods mentioned previously (including SA-EEND [179]). The amount of overlapped speech is around 13% of total speech duration.

6.3.2 Architecture, Training and Inference Details

We employ our Asteroid toolkit [262] to experiment with 2 SSep architectures: Conv-TasNet and DPRNN, both in online (causal) and offline (non-causal) configurations (for a total of 4). For both, we use the best hyperparameter configuration as found in [54, 176] with these exceptions: to reduce memory footprint we employ a 16 analysis/synthesis kernel size for encoder/decoder also for DPRNN and, regarding causal models, we use standard layer normalization versus the non-causal global layer normalization employed in non-causal models. Additionally, we set the DPRNN chunk and hop sizes to 100 and 50, respectively. These models are trained on the simulated fully overlapped Fisher dataset using the SI-SDR objective to separate two speakers. We use Adam optimizer [263], batch size 4 and learning rate 0.001. We clip gradients with l_2 norm greater than 5. Learning rate is halved if SI-SDR does not improve on validation for 10 epochs. If no improvement is observed for 20 epochs, training is stopped. Each SSep model is then fine-tuned using a learning rate of 0.0001 and batch size 1 on the real Fisher data, by taking 60 s long random segments from each recording. We tried to train the models from scratch using the real-world Fisher data directly but we failed. Since the speech is sparse, the models where prone to not separate at all and the training was excessively slow. Hence the use of the simulated fully overlapped data in order to “bootstrap” the training.

As said, we adopt the TCN VAD from [128], which is causal and for which the latency amounts to 10 ms. This model is trained here on the original Fisher data, using each speaker source separately, as the VAD is then applied only to separated sources. We train on random 2 s long segments with a batch size of 256. The rest of training hyperparameters are the same as those used for SSep models. During inference we employ a median filter to smooth the VAD predictions. In addition, we remove segments shorter than a threshold t_s to further reduce false alarm errors. For each SSGD model, we tune the median filter, leakage removal threshold and t_s parameters on the Fisher validation set (CALLHOME adaptation set for CALLHOME models). The segment length L of the leakage removal algorithm is set to 10 ms, which results in the same latency as the TCN VAD.

Table 6.1: Speech separation and diarization results on the Fisher and CALLHOME test sets in the **online** scenario. Separation is assessed using the SI-SDR (dB) improvements over the input mixtures. Diarization is assessed using diarization error rate (DER), missed speech (MS), false alarm (FA) and speaker confusion errors (SC). Latency of the system is reported in seconds. The best results among proposed techniques are shown in **bold**, and among EEND methods are underlined.

Method	**VAD**	**Latency**	**Fisher**					**CALLHOME**			
		(s)	**SI-SDRi**	**MS**	**FA**	**SC**	**DER**	**MS**	**FA**	**SC**	**DER**
SA-EEND+STB [249]		1									12.5
BW-EDA-EEND [250]		10									11.8
SA-EEND-EDA+STB [265]		10									10.0
Oracle sources	Energy		∞	7.4	1.4	0.1	8.9				
Oracle sources	TCN		∞	3.2	1.7	0.1	5.0				
Conv-TasNet	Energy	0.01	-0.9	11.5	39.1	9.5	60.1	7.3	55.8	5.6	68.7
Conv-TasNet	TCN	0.01	-0.9	1.7	70.3	2.2	74.1	**3.4**	82.3	0.6	86.4
+ Leakage removal	TCN	0.01	-3.1	5.2	5.6	25.9	36.8	6.2	21.9	15.5	42.6
DPRNN	Energy	0.1	**22.6**	7.6	**1.4**	**0.8**	9.7	5.5	6.9	1.9	14.3
DPRNN	TCN	0.1	**22.6**	**3.8**	2.6	**0.8**	7.1	5.9	4.5	**1.6**	12.0
+ Leakage removal	TCN	0.1	22.2	4.3	1.8	**0.8**	**6.8**	6.9	**2.3**	1.9	**11.1**

6.4 Experimental Analysis

We evaluate the performance on Fisher and CALLHOME test sets in terms of diarization error rate (DER) including overlapped speech and using a collar tolerance of 0.25 s, as in [179]. The evaluation is carried out using the standard NIST *md-eval* scoring tool [264]. For the Fisher test set we also report the SI-SDRi [260] source separation metric since oracle sources are available.

6.4.1 Online Separation/Diarization

The results for online SSGD diarization models are reported in Table 6.1. Oracle sources refers to SSGD with oracle SSep, thus with error coming only from the VAD module. We carry out the oracle evaluation only for Fisher, as for CALLHOME separated sources are not provided. For the CALLHOME evaluation, we also show DERs obtained by EEND, as reported in the original papers.

We observed that the Conv-TasNet model failed to deal with long recordings, generating large false alarm errors. This is due to the fact that, being fully convolutional, it has a limited ∼1.5 s receptive field. On the other hand, the DPRNN, being based on recurrent neural networks, has no such limitations and was effectively able to track the speakers for much longer and generate better diarization results. The proposed leakage removal algorithm was highly effective for both architectures. This was especially true in the case of TCN-based VAD since it was more prone to false alarms caused by leaked speech due to being trained on real Fisher data and not on the output of the separators. Although the algorithm was only partially able to mitigate the low separation capability of the Conv-TasNet, it improved the DER by 50.3% and 50.7% on Fisher and CALLHOME, respectively. For DPRNN, the improvement was lower as the system without leakage removal was already able to obtain good diarization performance. However, the proposed post-processing almost halved the false alarm error rates and improved the DER by 4.2% and 7.5% on Fisher and CALLHOME, respectively.

As a comparison, the current best performing online system on the CALLHOME dataset (i.e., SA-EEND-EDA with speaker tracing buffer [265]), obtains 10.0% DER, which is slightly better than ours but is obtained with significantly higher latency of 10 s. Our approach works with a latency of 0.1 s, making it appealing for applications where real-time requirements are very important (e.g., real-time captioning). Last but not least, the SSGD is trained using a dataset of ∼900 hours of speech, which is considerably smaller than the ones used to train the state-of-the-art EEND models (i.e., ∼10000 hours) and results in shorter training times and less burden regarding additional costs for the generation of simulated mixtures.

6.4.2 Offline Separation/Diarization

For the offline scenario, we compare our approach with clustering-based and EEND methods. For the former, we use VBx [266] and spectral clustering [225], along with their overlap-aware counterparts [170, 233]. For VAD in these systems, we use the publicly available Kaldi ASpIRE VAD model [267][1]. For overlap detection, we fine-tune the Pyannote [147] segmentation model[2] on the full CALLHOME adaptation set. The hyperparameters for each task are tuned on the corresponding validation set. The scripts for reproducing the baseline results are publicly available [3]. For fair comparison, we also report the performance of VBx with the TCN VAD, which however leads to degraded performance for this system.

The results for baselines and the offline SSGD diarization models are reported in Table 6.2. As in Table 6.1, we show DERs of the EEND methods for the CALLHOME test set.

In contrast to the online scenario, Conv-TasNet obtained good separation capability. However, DPRNN-based SSGD strongly outperformed the Conv-TasNet version on all metrics on the Fisher dataset, and even surpassed the

[1] `https://kaldi-asr.org/models/m4`

[2] `https://huggingface.co/pyannote/segmentation`

[3] `https://github.com/desh2608/diarizer`

Table 6.2: Speech separation and diarization results on the Fisher and CALLHOME test sets in the **offline** scenario. The best results among proposed techniques are shown in **bold**, and those among baselines are underlined. *Oracle sources* evaluation is the same of Table 6.1, as the VADs works online in both online and offline scenarios.

Method	**VAD**	**Fisher**					**CALLHOME**			
		SI-SDRi	**MS**	**FA**	**SC**	**DER**	**MS**	**FA**	**SC**	**DER**
VBx [266]	TCN		10.0	0.3	0.5	10.8	7.3	1.9	3.1	12.3
VBx [266]	Kaldi		8.9	0.4	0.9	10.2	8.3	0.9	2.6	11.7
+ Overlap assignment [170]	Kaldi		4.4	2.1	0.9	7.4	5.3	2.5	2.4	10.3
Spectral clustering [225]	Kaldi		8.9	0.4	0.2	9.5	8.3	0.9	5.3	14.5
+ Overlap assignment [233]	Kaldi		5.2	2.0	0.2	7.4	5.7	2.7	5.8	14.1
SA-EEND [179]										9.5
SA-EEND-EDA [244]										8.1
EEND + VC [245]							4.0	2.4	0.5	7.0
DIVE [255]										6.7
Conv-TasNet	Energy	17.5	8.0	4.5	1.6	14.1	6.0	12.0	2.8	20.6
Conv-TasNet	TCN	17.5	6.2	5.0	1.1	12.4	6.1	13.6	1.8	21.6
+ Leakage removal	TCN	17.1	5.5	2.5	2.0	10.1	6.0	10.1	2.8	18.9
DPRNN	Energy	**22.6**	7.6	**1.2**	**0.7**	9.5	5.5	4.4	0.5	10.4
DPRNN	TCN	**22.6**	**3.4**	2.2	**0.7**	6.3	**5.0**	5.4	**0.4**	10.8
+ Leakage removal	TCN	22.2	3.9	1.6	**0.7**	**6.1**	6.6	**1.9**	0.7	**9.3**

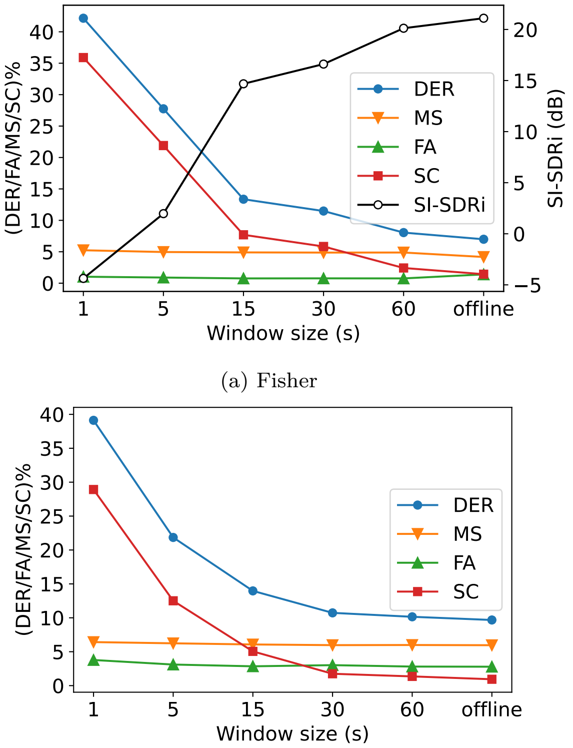
overlap-aware VBx which scored best among all clustering baselines. Regarding separation performance (SI-SDRi), we can see that the offline DPRNN did not improve over the online one. In general, the TCN VAD outperformed the energy-based one, especially when the former was used jointly with the proposed leakage removal, which continued to be effective in the offline configuration.

For the CALLHOME data, the best performing SSGD model is comparable with SA-EEND [179]. Although the diarization capability is good, it is not competitive with the current best performing approaches [245, 255], making it less attractive for offline applications. However, as we show in Section 6.4.4, it can be a more cost effective solution as the separated signals can be readily used in downstream applications such as ASR.

In future work we will consider several strategies to reduce this gap such as training with more data, comparable to the amount used in EEND models, and fine-tuning our models on the CALLHOME adaptation set (as done in [179, 244]).

6.4.3 CSS Window Analysis

Recall from Section 6.2.1 that the CSS framework, besides allowing the processing of arbitrarily long recordings, also allows to use a non-causal separation model in an online fashion with latency reduced to the length of the CSS window. Therefore, it can be regarded as an alternative approach for performing diarization online. We use the best SSGD offline model from Table 6.2 (DPRNN+TCN+Leakage removal) to investigate the effect of varying window sizes on SSGD. Evaluation results are reported in Fig. 6.3 for both datasets. As expected, the DER consistently decreased as the window size increased. In particular, the performances were almost on par with the offline models for windows larger than 60 and 30 seconds, respectively, for Fisher and CALLHOME. This suggests a possible parallelization scheme for offline SSGD by applying CSS on minute-long frames simultaneously, resulting in significant inference speed-ups and less memory consumption. The optimal chunk sizes are different for the two datasets because of the difference in their average recording duration (which is 10 minutes and 72 seconds for Fisher and CALLHOME, respectively).



(a) Fisher

(b) CALLHOME

Figure 6.3: Separation and diarization results on the test sets with different CSS windows. The overlap between windows is set to 50%. The results are obtained with the DPRNN+TCN+Leakage removal model.

As the window was shortened, missed speech and false alarm error rates remained approximately constant while speaker confusion errors consistently increased, indicating that the main source of error comes from speaker permutation due to wrong channel reordering during the stitching stage of the CSS. For smaller windows, the cross-correlation used for reordering consecutive chunks is less reliable due to the smaller size of the overlapping portion.

The CSS framework is not competitive with the online approach with causal SSep (Sec. 6.4.1) in terms of latency. However, it could be a convenient choice for applications in which better diarization accuracy is more desirable than the low-latency requirement, and memory footprint is an important concern, especially for very long recordings (e.g., > 10 minutes).

6.4.4 Automatic Speech Recognition Evaluation

A great advantage of the SSGD framework over other diarization methods is that separated sources together with the segmentation provided by the VAD can be readily fed in input to a back-end ASR system. To investigate ASR performance, we feed to a downstream ASR the estimated sources for the DPRNN models with and without leakage removal and using oracle segmentation or not. We use the pre-trained Kaldi ASPiRE ASR model [95][4] and report the performance in terms of word error rate (WER). We compare the results with the ones obtained with input mixtures and oracle sources, which ideally represent the upper and the lower bound for WER evaluation.

The results are reported in Table 6.3. We can see that for all SSGD systems the degradation was small compared to using oracle signals. This suggests that the separation is highly effective. A large improvement was obtained over the mixture, and we can observe that the main source of performance degradation versus a fully oracle system (oracle VAD + oracle sources) comes from the VAD segmentation. This is consistent with what we observed for diarization in Sections 6.4.1 and 6.4.2. The leakage removal algorithm slightly degraded the performance, but, on the other hand, in the proposed framework it could be only used for obtaining the segmentation and avoided for ASR (*+ Leakage removal (seg-only)*). In this latter case the performance was slightly increased.

6.5 Conclusion & Future Work

In this Chapter we gave a brief historical overview of the field of speaker diarization, including the most recent directions which have a strong focus on better handling of overlapped speech.

Following we presented our study [203] on the use of SSGD for real-world telephone conversations. In this work we extended SSGD to online diarization

[4] `https://kaldi-asr.org/models/m1`

Table 6.3: WER evaluation on the Fisher test set. The best online/offline non-oracle results are reported in **bold**.

Method	**Online**	**VAD**	
		TCN	**Oracle**
Mixture		38.74	30.69
Oracle sources		25.44	19.50
DPRNN	✓	26.42	20.89
+ Leakage removal	✓	26.94	21.03
+ Leakage removal (seg-only)	✓	**26.21**	n.a.
DPRNN	✗	26.21	21.13
+ Leakage removal	✗	26.68	21.26
+ Leakage removal (seg-only)	✗	**26.13**	n.a.

scenarios and arbitrarily long audio streams. We have shown that our best online SSGD system is able to achieve comparable performance with state-of-the-art methods based on EEND on the CALLHOME dataset with significantly lower latency (for instance, 0.1 s compared to 10 s). It also exhibits overall stronger performance than state-of-the-art clustering methods even in their overlap-aware variant. We considered also the use of CSS with non-causal separation models and how this could impact downstream diarization performance. Our findings suggest that DERs were almost on par with the offline case with a sufficiently large CSS window of 60 or 30 seconds for Fisher and CALLHOME datasets, respectively. Finally, we have shown that SSGD is particularly appealing for multi-talker speaker-attributed ASR, since the estimated sources could be fed directly to an ASR module, leading to significant ASR performance boost. Future work could investigate joint fine-tuning of separation and VAD to reduce these errors, e.g. on the CALLHOME adaptation set. Another direction is to extend the SSGD framework performance to domains other than CTS (e.g., meeting-like and dinner scenarios) where an higher number of speakers could be involved. This however requires the development of new techniques since most current source separation methods struggle to track 3 or more speakers for very long inputs.

Chapter 7

Conclusions

In this dissertation we addressed various aspects of front-end speech processing and proposed various algorithms for the tasks of multi-channel speech enhancement, channel selection, keyword spotting, speaker counting and speech separation driven diarization. As said, our main focus was on deep learning driven techniques, which have become the de-facto standard approach for many of these tasks. Such techniques are considered computational intensive compared with more classical machine learning and classical DSP approaches. For this reason particular attention has also been given to the computational aspect in the works presented here, and, in some instances also on low algorithmic latency. This is an often neglected aspect but it is of utmost importance in practical applications, and is even more crucial for applications that have to run on edge-devices, as the front-end pre-processing is done on-device usually. An higher computational cost often means a lower budget for other on-device parallel applications, as well as an higher impact on energy consumption, leading to worse battery life and/or to environmental concerns [18, 19, 268].

To allow for efficient DNN-based front-end processing algorithms, in the works presented in this dissertation, we had to develop novel techniques and methodologies. For example regarding the DNN architecture (as seen e.g. in Chapter 5 for the transformer-based network), or again, regarding the development of novel algorithms (as the leakage removal algorithm in Chapter 6), or even the invention of completely novel frameworks for channel selection (Chapter 3), acoustic echo cancellation (Chapter 4) and acoustic beamforming (Chapter 2). We summarize our contributions more in detail in the following.

7.1 Summary of Contributions

In Chapter 2: we presented a work on multi-channel speech enhancement via DNN-supported classical beamforming. In this framework, a single-channel DNN is used to estimate a magnitude STFT mask for the target speaker and the interferers/noise; this mask is then exploited to compute conventional beamforming solutions such as MWF and MVDR which are then used to enhance the input signal. The novelty of our work is the fact that we also explore learned domains beyond the STFT. This is possible by learning suitable analysis and synthesis filterbanks together with the mask estimation DNN.

Our experiments, performed on the First Clarity Enhancement dataset [61], shows that in some instances and MVDR solution by using learned filterbanks is possible to consistently outperform STFT-based systems, even when oracle-based masks are employed. We found out that such performance improvement is generally afforded by the use of an over-complete basis, with the best results found for filterbanks with small kernel size compared to the number of filters. This reflects findings on encoder-masker-decoder end-to-end monaural speech separation, which has relied on this principle since TasNet [269] and Conv-TasNet [176].

In Chapter 3: we presented our MicRank framework, which allows for fully neural data-driven channel selection. We propose to frame the channel selection problem as a learning to rank (LTR) problem and several ranking strategies are compared, notably RankNet and ListNet. The model is designed to be lightweight and suitable for edge-devices deployment.

Results on synthetic data show that this approach is able to surpass considerably previous techniques and even some oracle-based measures. However further work is needed to address successfully multi-speaker data, due to the presence of speech overlap. On such scenario the gap with respect to oracle measures, remains large, but it still outperforms previously proposed blind channel selection approaches.

As an additional contribution we also studied the effect of envelope variance (EV) channel selection when used in combination with GSS in the context of the CHiME-7 DASR challenge. The results show that channel selection is able to improve the separation stage in some scenarios and, in all scenarios, reduce the computational requirements.

In Chapter 4: a novel framework called implicit acoustic echo cancellation was presented, and validated experimentally for the arduous tasks of on-device continuous keyword spotting (KWS) and device-directed speech detection (DDD) using both a synthetic dataset and a real-world Alexa device dataset. This framework tackles the “barge-in” problem, which consists in the user voice overlapping with the device playback sound, usually a TTS response. Since the device playback signal is known a-priori we propose to feed it as an additional feature to the KWS or DDD classifier such that it can learn to disambiguate between the use and the TTS response and ignore this latter. This technique leads to two order of magnitude less computational requirements than competing approaches. A contrastive, on-the-fly data augmentation technique is also developed, which allows to reduce potential biases (e.g. perceived TTS model gender) in the data.

In Chapter 5: we propose to treat voice-activity detection (VAD), overlapped speech detection (OSD) and speaker counting in an unified way under an unique novel overlapped speech detection and counting (OSDC) framework. Several DNN architectures are compared, with a particular attention to computational complexity and two novel architectures are proposed one based on TCN and another based on Transformer. We validate them on real-world meeting scenarios captured by array devices (AMI and CHiME-6) and compare results obtained with speaker counting, OSD and VAD training targets. We also conduct experiments on the use of spatial features to aid in these tasks and conclude these can bring significant improvement especially when employed in a late-fusion fashion.

In Chapter 6: a study on speech-separation guided diarization (SSGD) for conversational telephone speech is presented. The main focus is on low-latency diarization, we compare two state-of-the-art separation models Conv-TasNet and DPRNN, both causal and non-causal and also with/without the continuous speech separation framework. The two popular telephone diarization CALLHOME and Fisher datasets are considered. A novel, simple and very effective leakage removal algorithm is proposed for the SSGD framework, and it is shown it can considerably boost the performance without significant computational overhead. Results outline that SSGD is competitive with current state-of-the-art diarization methods in the low-latency scenario, and can achieve near perfect separation in telephone conversational data, leading to significant improvements for applications such as ASR.

7.2 Current Trends and Possible Future Directions

7.2.1 End-to-End Integration and Modularity

One common theme in some of the works presented in this dissertation, is the integration between different front-end speech processing aspects, such as localization and speaker counting, speech separation and diarization or acoustic echo cancellation and keyword spotting. This is a more general trend in the speech processing field right now. This trend has been made possible largely by the adoption of data-driven deep learning techniques. Such methods naturally allow for end-to-end optimization when the blocks are cascaded together, leading to large improvements in terms of performance as well as new opportunities related to the fact that data from different domains can be leveraged. For example, regarding SSE, end-to-end optimization with ASR, it is possible to reduce the reliance on synthetic data, as the oracle target signal is not anymore needed for the fine-tuning part, but only transcriptions if the ASR loss only is used. Several works explore this direction and reported clear performance gains in doing so [25, 58, 270–274]. As said, integration is a general trend and significant efforts are ongoing for e.g. integration between ASR and diarization [275, 276], diarization and SSE [203, 237, 258, 277] or as seen also between ASR and SSE (ASR can also help SSE [278]). Our recent contribution, the ESPNet-SE++ [25] toolkit for example, is meant to help explore such integration possibilities between SSE and different downstream applications, not limited to ASR.

Related to the concept of end-to-end integration is also the concept of modularity. Complete black-box fully neural end-to-end methods that perform implicitly different tasks e.g. beamforming within ASR [57, 279, 280] are appealing, They can be optimized directly for a given specific task without the additional burden of using simulated data for training the front-end. However, we argue that a modular approach has several advantages even if it may not reach the same level of performance on that very same task/domain due to lower capacity. First, they are more interpretable compared to fully neural methods, as the output of each different component is well defined, as it belongs to a given sub-task e.g. SSE and ASR. Second, a modular system components may be swapped e.g. depending on the domain. Single components are reusable. Third, they can possibly leverage more data as a whole, since different components may require different heterogeneous training data as for each task the annotation can differ. Fourth, if designed well they can still allow for fine-tuning on a given domain, thus reaching very close performance to end-to-end black-box approaches, and possibly, even surpass these latter due to the capability of leveraging data from different heterogeneous sources. One great example of this is the recently proposed LegoNN approach [281] which proposes a end-to-end but modular ASR framework where language components and acoustic components can be "swapped" based on the language and the domain. It is easy to see that this could also allow to leverage more data as a whole, as each module can be trained on different corpora which may not be suitable for other modules. Another example is our recent MultiIRIS [58] work, in which we explored the integration of beamforming, ASR and also self-supervised learning representation (SSLR) features. We report here, for convenience, in Figure 7.1 the block scheme of such work.

In this recent work we demonstrate how an hybrid DNN-based beamforming front-end based on a weighted power minimization (WPD) [282] solution could be used to boost ASR performance in noisy-reverberant environments even when this latter is quite strong. The ASR model employs WavLM [15] as the input feature extractor, and it thus leverages tens of thousands of training data used in the WavLM pre-training. It is also trained on multi-condition, both on clean and corrupted utterances. With joint fine-tuning of the ASR back-end with the front-end, we found that one can considerably improve the

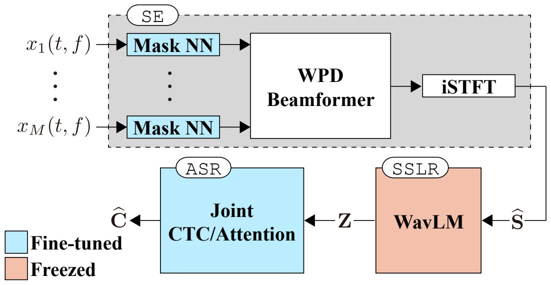


Figure 7.1: Integration between beamforming, ASR and self-supervised learning representation (SSLR), as proposed recently in MultiIRIS.

performance not only of the ASR back-end but also, surprisingly of the front-end, as the objective signal-based metrics also increase. This latter, via fine-tuning, can now leverage also real-world data for which the oracle clean speech targets are not available.

In our opinion we will see more works following these paradigms of integration and modularity, and maybe even on the line of the "network of neural networks" paradigm proposed by Ravanelli [283] scaled to several speech processing tasks, e.g. SSE, diarization and ASR with each task/module helping refining the predictions of the other.

7.2.2 Efficiency

Another recurrent theme in this dissertation is computational efficiency. Deep learning has proven to be extremely effective and has enabled a significant leap in performance as we discussed in the Introduction. However it is undoubted that some of these results also came to the detriment of important concepts such as efficiency and parsimony.

This is exemplified for example by the field of speech/source separation, where in the last few years the ongoing trend has been to design new methods each time more computational intensive in terms of FLOPs; this to beat the state-of-the-art on the popular WSJ0-2mix benchmark dataset [284]. Undoubtedly, this research direction brought significant improvements and new ideas and very effective neural architectures such as Conv-TasNet, DPRNN and SepFormer. But it is also true that the performance on such dataset has saturated for the past 3 years at least. The SI-SDR is so high that the results have become almost indistinguishable from the oracle targets. Some exceptions, to be fair, exists such as SuDoRM-RF [285]. The same trend is observed on ASR and has been pointed out by Parcollet et al. [18]. By focusing only on improving a particular metric a few decimal points, on a over-used dataset we may be missing the bigger picture. Instead the focus should shift more, in our humble opinion, towards more ambitious goals such as generalization across real-world speech-in-the-wild domains and computationally efficient methods.

This is not to say that all efforts in the direction of pushing the state-of-the-art are wasted resources. But that some little additional effort, such as also reporting results on more up-to-time datasets, and/or reporting the number of FLOPs or multiply-accumulate operations should be encouraged more and more.

After all large scale models trained on massive amounts of data are undoubtedly useful and crucial to advance in the field. Whisper [76] is a good example of this and we are sure there will be more works that go into the direction of performing quantization and/or knowledge distillation with such large scale models for the purpose of obtaining an efficient and more practically viable ASR system that can be deployed, while retaining robustness.

Another thing we researchers can do is to try to push more for efficiency by organizing challenges and special sessions that focus also on this aspect. An example is our recently organized DCASE Task 4 2022 challenge [1], where we also explored a new requirement for the participants to also report energy consumption at training and test time. Or our recent ICASSP special session[2] focused on resource-efficient real-time neural speech separation. Through this special sessions we are trying to foster research towards practically viable SSE algorithms that can lead to deployment in real-world products and have a tangible impact in the everyday world.

Last but not least, efficiency may also hold the key for true artificial intelligence, after all our brain is orders of magnitude more efficient compared to current deep learning algorithms.

7.2.3 Limits of this Dissertation

Probably the single most important missing argument in this dissertation is self-supervised learning. As mentioned in the Introduction, in the last two years, from roughly Wav2vec 2.0 [13] on-wards (but the trend was already visible before, at least in ASR), the use of self-supervised learning (SSL) techniques is changing the field of speech processing and more and more research is been done in this direction. Regarding the field of front-end speech processing, large-scale pre-trained SSL models such as Wav2Vec 2.0 and WavLM, have been demonstrated to be very effective for example in KWS [286] and diarization [15].

On the other hand, they have been found to be less effective on other tasks, for example speech separation and enhancement [15]. For these two latter tasks, in recent years purposely tailored SSL methods have been developed,

[1] dcase.community/challenge2022/task-sound-event-detection-in-domestic-environments

[2] 2023.ieeeicassp.org/detailed-presentation-schedule/

such as MixIT [12] and its variants such as RemixIT [287], Self-Remixing [288] as well as SAMoM [289] for target-speaker extraction. These methods have been proven to achieve, on synthetic data, performance comparable in some cases to fully supervised techniques. However how useful MixIT and its variants can be for e.g. boosting ASR performance in real-world applications such as meeting transcription, is still an open question. On the contrary, there is little doubt about the capability of ASR-targeted SSL to improve performance on real-world data. There are already some encouraging works [290] about the use of MixIT for semi-supervised speech separation. Another example is the ongoing CHiME-7 UDASE Challenge[3] which focuses exactly on this issue and employs RemixIT as the baseline approach. On our side, we successfully employed unsupervised universal sound separation via MixIT to improve sound event detection performance (SED) for the DCASE 2021 Task 4 baseline [4]. By leveraging a model trained with MixIT in a totally unsupervised way on the YFC100m [291] dataset, we were able to considerably boost the SED performance on the evaluation set. The joint separation+SED model generalized better as it could leverage orders of magnitude more data due to the unsupervised pre-training of the front-end separation module. Again, the leitmotifs of integration and modularity explained before are also valid for this work.

Another open question, especially for what regards large-scale SSL models such as WavLM is how to leverage them, in the best possible way when multichannel data is available. Such research question is still open and is one of the focuses of the ongoing CHiME-7 DASR Challenge[5] for example.

It is must be also said that, most works that explore the use of SSL, have been focusing largely on raw performance alone, without too much consideration for efficiency. As explained before efficiency is a must-have in many front-end speech processing applications. Some exceptions however exists, especially for KWS [292], as well as ASR [293–295]. Some of these works employ knowledge distillation to reduce the computational requirements at run-time by distilling a large model into a smaller one. However, while this technique is well developed and proven for classification tasks, it is an open question if it can be also effective for tasks such as speech separation.

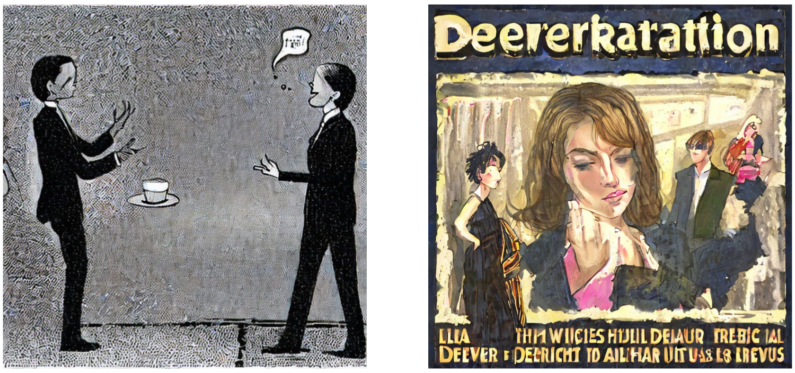
Another very recent "hot topic" missing in this dissertation, at the time of this writing, are generative methods. So-called large language models (LLM) such as ChatGPT have been demonstrating impressive capabilities in generating coherent and contextually relevant text, from news articles and creative writing to automated dialogue systems. LLMs have opened up exciting possibilities for natural language generation and have the potential to revolutionize

[3] chimechallenge.org/current/task2/index

[4] dcase.community/challenge2021/task-sound-event-detection-and-separation-in-domestic-environments

[5] chimechallenge.org/current/task1/index

many fields, from assistive technologies to content creation. In the last year we have also assisted in a significant stride in the field of conditional image generation. Models such as DALL-E [296] and, more recently ones based on stable diffusion [297], can generate high-quality realistic images from textual descriptions, effectively bridging the gap between natural language and visual information. They are also undoubtedly fun to use as Figure 7.2 demonstrates, with often unexpected and interesting results.


Deererkaratiion
LLIA THM WIICIES HIJLIL DEIAUR TREBIC IAL
DEEVER DECRICHT TO AILHAR UIT U4S L8 LREVUS

(a)

(b)

Figure 7.2: An image generated using stable diffusion via the Huggingface Stable Diffusion 2.1 API when prompted with (a) “speech separation” and (b) “dereverberation”. The result is quite “Kafkaesque” in the left one.

These technologies are also impacting the audio field as the recent works done in the past months demonstrate. For example, in the direction of conditional audio generation [298, 299], music generation [300], text-to-speech [301], spoken dialogues [302], and even audio neural encoding [303], as well as multi-modal systems able to process audio and text seamlessly [304]. Potential applications include more principled synthetic data generation. For example, regarding audio applications, high-quality audio scene synthesis could allow for better robustness for front-end and back-end audio applications (as more data can be leveraged), less mismatch with real-world data and more crucially, privacy preservation. This latter aspect is crucial for domains where data is especially sensitive such as doctor-patient meetings, industry (e.g. trade secrets) as well as in government and international institutions, just to name a few. In these domains data could be very scarce or not even available at all due to being extremely sensitive. Automatically generating fake synthetic data in such domains then could be extremely effective, and promising results have been already produced for example in the field of data analytics [305, 306]. We believe that these approaches could be be very well extended also to the audio field by leveraging these new powerful techniques, thus allowing the training of the models on “proxy” fake data, with minimal mismatch with respect to the target domain.

To be fair, it must be also said that, as with any rapidly advancing technology, there are also concerns about the ethical implications of such powerful tools, including the potential for misuse e.g. identity theft or fake news creation (so-called “deep-fakes”). As we continue to develop these approaches it is thus also crucial to consider the ethical implications and ensure that these technologies are used to the benefit of society and not to its detriment, as instruments of oppression.

7.3 Final Remarks

As said in the Introduction, it is an especially exciting time to be able to work in this field due to the astounding progress made in recent years. With access to vast amounts of data and powerful computing resources, we are seeing breakthroughs well beyond the realm of speech processing: computer vision, drug discovery, natural language processing and so on. However, we also believe that in the next years we will have also to address some of the main limitations of deep learning in order to get further advancements. For one, as we already mentioned, it is far from being efficient, current state-of-the-art models that go into newspaper headlines such as ChatGPT or Whisper [76], require massive resources to be trained. Whether such approach could scale much further, and whether noticeable gains will be reached by doing this, would be likely found in the next couple of years, but there is already indication of diminishing returns [18].

We believe that one main ingredient that is missing from the current mainstream approach is the concept of feedback and long-term memory. Humans learn by interacting with the environment in a continuous manner. Instead the current approaches are mostly based on stochastic gradient descent and mini-batch training. During training each example is assumed independent and identically distributed amongst the dataset and the model weights are adjusted for each mini-batch independently. There is no explicit memory of the past examples in the current learning paradigm, leading e.g. to phenomena such as catastrophic forgetting.

To be able to attain more flexible and robust machine learning algorithms we should strive to gradually move away from such assumptions and, inspired by biological systems, move towards more plausible and principled learning schemes.

Bibliography

[1] A.-M. Salai, A. Kirton, G. Cook, and L. Holmquist. “Views and experiences on the use of voice assistants by family and professionals supporting people with cognitive impairments”. In: *Frontiers in Dementia* 1 (2022), p. 1049464.

[2] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. “Learning representations by back-propagating errors”. In: *nature* 323.6088 (1986), pp. 533–536.

[3] K. Fukushima. “Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position”. In: *Biological cybernetics* 36.4 (1980), pp. 193–202.

[4] S. Hochreiter and J. Schmidhuber. “Long short-term memory”. In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[5] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[6] A. Krizhevsky, I. Sutskever, and G. E. Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Communications of the ACM* 60.6 (2017), pp. 84–90.

[7] V. Panayotov et al. “Librispeech: An ASR corpus based on public domain audio books”. In: *Proc. of ICASSP*. 2015, pp. 5206–5210.

[8] L. D. Consortium. *1997 HUB5 English Evaluation. Available at* `https://catalog.ldc.upenn.edu/LDC2002S23` (accessed September 14, 2022).

[9] A. Graves. “Sequence Transduction with Recurrent Neural Networks”. In: *International Conference on Machine Learning* (2012).

[10] A. Graves and A. Graves. “Connectionist temporal classification”. In: *Supervised sequence labelling with recurrent neural networks* (2012), pp. 61–93.

[11] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. “Attention is all you need”. In: *Advances in NIPS* 30 (2017), pp. 6000–6010.

[12] S. Wisdom, E. Tzinis, H. Erdogan, R. Weiss, K. Wilson, and J. Hershey. "Unsupervised sound separation using mixture invariant training". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 3846–3857.

[13] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli. "wav2vec 2.0: A framework for self-supervised learning of speech representations". In: *Advances in neural information processing systems* 33 (2020), pp. 12449–12460.

[14] W.-N. Hsu, B. Bolte, Y.-H. H. Tsai, K. Lakhotia, R. Salakhutdinov, and A. Mohamed. "Hubert: Self-supervised speech representation learning by masked prediction of hidden units". In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 29 (2021), pp. 3451–3460.

[15] S. Chen, C. Wang, Z. Chen, Y. Wu, S. Liu, Z. Chen, J. Li, N. Kanda, T. Yoshioka, X. Xiao, et al. "WavLM: Large-scale self-supervised pretraining for full stack speech processing". In: *IEEE Journal of Selected Topics in Signal Processing* 16.6 (2022), pp. 1505–1518.

[16] H. Xu, J. Li, A. Baevski, M. Auli, W. Galuba, F. Metze, C. Feichtenhofer, et al. "Masked autoencoders that listen". In: *Advances in neural information processing systems* (2022).

[17] S.-w. Yang, P.-H. Chi, Y.-S. Chuang, C.-I. J. Lai, K. Lakhotia, Y. Y. Lin, A. T. Liu, J. Shi, X. Chang, G.-T. Lin, T.-H. Huang, W.-C. Tseng, K.-t. Lee, D.-R. Liu, Z. Huang, S. Dong, S.-W. Li, S. Watanabe, A. Mohamed, and H.-y. Lee. "SUPERB: Speech Processing Universal PERformance Benchmark". In: *Proc. of Interspeech.* 2021, pp. 1194–1198.

[18] T. Parcollet and M. Ravanelli. "The energy and carbon footprint of training end-to-end speech recognizers". In: *Proc. of Interspeech* (2021).

[19] R. Schwartz, J. Dodge, N. A. Smith, and O. Etzioni. "Green AI". In: *Communications of the ACM* 63.12 (2020), pp. 54–63.

[20] T. Ochiai, S. Watanabe, and S. Katagiri. "Does speech enhancement work with end-to-end ASR objectives?: Experimental analysis of multichannel end-to-end ASR". In: *2017 IEEE 27th International Workshop on Machine Learning for Signal Processing (MLSP).* IEEE. 2017, pp. 1–6.

[21] K. Iwamoto, T. Ochiai, M. Delcroix, R. Ikeshita, H. Sato, S. Araki, and S. Katagiri. "How bad are artifacts?: Analyzing the impact of speech enhancement errors on asr". In: *arXiv preprint arXiv:2201.06685* (2022).

[22] Y. Qin, N. Carlini, G. Cottrell, I. Goodfellow, and C. Raffel. “Imperceptible, robust, and targeted adversarial examples for automatic speech recognition”. In: *International conference on machine learning*. PMLR. 2019, pp. 5231–5240.

[23] J. B. Li, S. Qu, X. Li, Z. Kolter, and F. Metze. “Real world audio adversary against wake-word detection systems”. In: *Proc. of NIPS*. 2019.

[24] E. Vincent, T. Virtanen, and S. Gannot. *Audio source separation and speech enhancement*. John Wiley & Sons, 2018.

[25] Y.-J. Lu, X. Chang, C. Li, W. Zhang, S. Cornell, Z. Ni, Y. Masuyama, B. Yan, R. Scheibler, Z.-Q. Wang, et al. “ESPnet-SE++: Speech enhancement for robust speech recognition, translation, and understanding”. In: *Proc. of ICASSP* (2022).

[26] M. Delcroix, K. Zmolikova, K. Kinoshita, A. Ogawa, and T. Nakatani. “Single Channel Target Speaker Extraction and Recognition with Speaker Beam”. In: *Proc. of ICASSP*. IEEE. 2018, pp. 5554–5558.

[27] M. Przybocki and M. Alvin. *2000 NIST Speaker Recognition Evaluation LDC2001S9*. 2001. URL: `https://catalog.ldc.upenn.edu/LDC2001S97`.

[28] C. Cieri, D. Miller, and K. Walker. “The Fisher corpus: A resource for the next generations of speech-to-text”. In: *LREC*. Vol. 4. 2004, pp. 69–71.

[29] S. Cornell, M. Pariente, F. Grondin, and S. Squartini. “Learning filterbanks for end-to-end acoustic beamforming”. In: *Proc. of ICASSP*. IEEE. 2022, pp. 6507–6511.

[30] M. Pariente, S. Cornell, A. Deleforge, and E. Vincent. “Filterbank design for end-to-end speech separation”. In: *Proc. of ICASSP*. 2020, pp. 6364–6368.

[31] J. Heymann, L. Drude, A. Chinaev, and R. Haeb-Umbach. “BLSTM supported GEV beamformer front-end for the 3rd CHiME challenge”. In: *Proc. of ASRU*. 2015.

[32] A. Aroudi and S. Braun. “DBNET: DOA-driven beamforming network for end-to-end farfield sound source separation”. In: *arXiv preprint arXiv:2010.11566* (2020).

[33] G. Li, S. Liang, S. Nie, W. Liu, Z. Yang, and L. Xiao. “Deep Neural Network-Based Generalized Sidelobe Canceller for Robust Multi-Channel Speech Recognition.” In: *Proc. of Interspeech*. 2020.

[34] C. Boeddeker, H. Erdogan, T. Yoshioka, and R. Haeb-Umbach. “Exploring Practical Aspects of Neural Mask-Based Beamforming for Far-Field Speech Recognition”. In: *Proc. of ICASSP*. 2018.

[35] X. Xiao, S. Zhao, D. L. Jones, E. S. Chng, and H. Li. “On time-frequency mask estimation for MVDR beamforming with application in robust speech recognition”. In: *Proc. of ICASSP*. 2017.

[36] J. Heymann, L. Drude, C. Boeddeker, P. Hanebrink, and R. Haeb-Umbach. “Beamnet: End-to-end training of a beamformer-supported multi-channel ASR system”. In: *Proc. of ICASSP*. 2017.

[37] X. Xiao, S. Watanabe, H. Erdogan, L. Lu, J. R. Hershey, M. L. Seltzer, G. Chen, Y. Zhang, M. Mandel, and D. Yu. “Deep beamforming networks for multi-channel speech recognition”. In: *Proc. of ICASSP*. 2016.

[38] H. Erdogan, J. R. Hershey, S. Watanabe, M. I. Mandel, and J. Le Roux. “Improved MVDR beamforming using single-channel mask prediction networks.” In: *Proc. of Interspeech*. 2016, pp. 1981–1985.

[39] T. Ochiai, S. Watanabe, T. Hori, J. R. Hershey, and X. Xiao. “Unified Architecture for Multichannel End-to-End Speech Recognition With Neural Beamforming”. In: *IEEE Journal of Selected Topics in Signal Processing* 11.8 (2017), pp. 1274–1288.

[40] T. Ochiai, M. Delcroix, R. Ikeshita, K. Kinoshita, T. Nakatani, and S. Araki. “Beam-TasNet: Time-domain Audio Separation Network Meets Frequency-domain Beamformer”. In: *Proc. of ICASSP*. 2020.

[41] Z. Zhang, Y. Xu, M. Yu, S. Zhang, L. Chen, and D. Yu. “ADL-MVDR: All deep learning MVDR beamformer for target speech separation”. In: *Proc. of ICASSP*. 2021.

[42] Y. Luo, C. Han, N. Mesgarani, E. Ceolini, and S. Liu. “Fasnet: Low-latency adaptive beamforming for multi-microphone audio processing”. In: *Proc. of ASRU*. 2019.

[43] Y. Luo, Z. Chen, N. Mesgarani, and T. Yoshioka. “End-to-end microphone permutation and number invariant multi-channel speech separation”. In: *Proc. of ICASSP*. 2020.

[44] Y. Luo and N. Mesgarani. “Implicit filter-and-sum network for multi-channel speech separation”. In: *arXiv preprint arXiv:2011.08401* (2020).

[45] Y. Xu, Z. Zhang, M. Yu, S. Zhang, and D. Yu. “Generalized Spatio-Temporal RNN Beamformer for Target Speech Separation”. In: *Proc. of Interspeech* (2020).

[46] X. Li, Y. Xu, M. Yu, S. Zhang, J. Xu, B. Xu, and D. Yu. “MIMO Self-attentive RNN Beamformer for Multi-speaker Speech Separation”. In: *Proc. of Interspeech* (2021).

[47] J. Capon. “High-resolution frequency-wavenumber spectrum analysis”. In: *Proceedings of the IEEE* 57.8 (1969), pp. 1408–1418.

[48] E. Warsitz and R. Haeb-Umbach. “Blind Acoustic Beamforming Based on Generalized Eigenvalue Decomposition”. In: *IEEE Transactions on Audio, Speech, and Language Processing* 15.5 (2007), pp. 1529–1539.

[49] J. Qi, H. Hu, Y. Wang, C.-H. H. Yang, S. M. Siniscalchi, and C.-H. Lee. “Exploring Deep Hybrid Tensor-to-Vector Network Architectures for Regression Based Speech Enhancement”. In: *Proc. of Interspeech.* 2020.

[50] Y. Fu, J. Wu, Y. Hu, M. Xing, and L. Xie. “DESNet: A Multi-Channel Network for Simultaneous Speech Dereverberation, Enhancement and Separation”. In: *Proc. of SLT.* 2021.

[51] Y.-J. Lu, S. Cornell, X. Chang, W. Zhang, C. Li, Z. Ni, Z.-Q. Wang, and S. Watanabe. “Towards low-distortion multi-channel speech enhancement: The ESPNet-SE submission to the L3DAS22 challenge”. In: *Proc. of ICASSP.* 2022, pp. 9201–9205.

[52] Z.-Q. Wang, S. Cornell, S. Choi, Y. Lee, B.-Y. Kim, and S. Watanabe. “TF-GridNet: Integrating Full-and Sub-Band Modeling for Speech Separation”. In: *Proc. of ICASSP* (2022).

[53] Y. Luo and N. Mesgarani. “Conv-TasNet: Surpassing Ideal Time–Frequency Magnitude Masking for Speech Separation”. In: *IEEE/ACM transactions on audio, speech, and language* 27.8 (Aug. 2019), pp. 1256–1266.

[54] Y. Luo, Z. Chen, and T. Yoshioka. “Dual-Path RNN: efficient long sequence modeling for time-domain single-channel speech separation”. In: *Proc. of ICASSP.* IEEE. 2020, pp. 46–50.

[55] C. Boeddeker, J. Heitkaemper, J. Schmalenstroeer, L. Drude, J. Heymann, and R. Haeb-Umbach. “Front-end processing for the CHiME-5 dinner party scenario”. In: *Proc. of CHiME-5 Workshop on Speech Processing in Everyday Environments.* 2018, pp. 35–40.

[56] W. Zhang, C. Boeddeker, S. Watanabe, T. Nakatani, M. Delcroix, K. Kinoshita, T. Ochiai, N. Kamo, R. Haeb-Umbach, and Y. Qian. “End-to-End Dereverberation, Beamforming, and Speech Recognition with Improved Numerical Stability and Advanced Frontend”. In: *Proc. of ICASSP.* 2021.

[57] F.-J. Chang, M. Radfar, A. Mouchtaris, B. King, and S. Kunzmann. "End-to-end multi-channel transformer for speech recognition". In: *Proc. of ICASSP*. IEEE. 2021, pp. 5884–5888.

[58] Y. Masuyama, X. Chang, S. Cornell, S. Watanabe, and N. Ono. "End-to-End Integration of Speech Recognition, Dereverberation, Beamforming, and Self-Supervised Learning Representation". In: *Spoken Language Technology Workshop*. 2023, pp. 260–265.

[59] M. Souden, J. Benesty, and S. Affes. "On optimal frequency-domain multichannel linear filtering for noise reduction". In: *IEEE Transactions on audio, speech, and language processing* 18.2 (2009), pp. 260–276.

[60] M. Kowalski, E. Vincent, and R. Gribonval. "Beyond the narrowband approximation: Wideband convex methods for under-determined reverberant audio source separation". In: *IEEE Transactions on Audio, Speech, and Language Processing* 18.7 (2010), pp. 1818–1829.

[61] M. A. Akeroyd, J. P. Barker, T. J. Cox, J. Culling, S. Graetzer, G. Naylor, E. Porter, and R. Viveros Muñoz. "Launching the first "Clarity" Machine Learning Challenge to revolutionise hearing device processing". In: *The Journal of the Acoustical Society of America* 148.4 (2020), pp. 2711–2711.

[62] D. Schröder and M. Vorländer. "RAVEN: A real-time framework for the auralization of interactive virtual environments". In: *In Proceedings of Forum Acusticum*. 2011.

[63] J. Le Roux, S. Wisdom, H. Erdogan, and J. R. Hershey. "SDR–half-baked or well done?" In: *Proc. of ICASSP*. 2019.

[64] D. P. Kingma and J. Ba. "Adam: A Method for Stochastic Optimization". In: *arXiv preprint arXiv:1412.6980* (2014).

[65] E. Vincent, R. Gribonval, and C. Févotte. "Performance measurement in blind audio source separation". In: *IEEE transactions on audio, speech, and language processing* 14.4 (2006), pp. 1462–1469.

[66] S. Cornell, A. Brutti, M. Matassoni, and S. Squartini. "Learning to rank microphones for distant speech recognition". In: *Proc. of Interspeech*. 2021.

[67] J. Barker, S. Watanabe, E. Vincent, and J. Trmal. "The Fifth 'CHiME' Speech Separation and Recognition Challenge: Dataset, Task and Baselines". In: *Proc. of Interspeech*. 2018, pp. 1561–1565.

[68] S. Watanabe et al. "CHiME-6 Challenge: Tackling multispeaker speech recognition for unsegmented recordings". In: *6th CHiME International Workshop on Speech Processing in Everyday Environments*. 2020.

[69] M. R. Bai, J. Ih, and J. Benesty. *Acoustic array systems: theory, implementation, and application.* John Wiley & Sons, 2013.

[70] X. Anguera. *Beamformit, the fast and robust acoustic beamformer.* 2006.

[71] J. Heymann, L. Drude, and R. Haeb-Umbach. "Neural network based spectral mask estimation for acoustic beamforming". In: *Proc. of ICASSP.* 2016, pp. 196–200.

[72] T. Yoshioka, Z. Chen, D. Dimitriadis, W. Hinthorn, X. Huang, A. Stolcke, and M. Zeng. "Meeting Transcription Using Virtual Microphone Arrays". In: *ArXiv* abs/1905.02545 (2019).

[73] X. L. Zhang. "Deep ad-hoc beamforming". In: *Computer Speech & Language* 68 (2021), p. 101201.

[74] Y. Luo, Z. Chen, N. Mesgarani, and T. Yoshioka. "End-to-end Microphone Permutation and Number Invariant Multi-channel Speech Separation". In: *Proc. of ICASSP.* 2020, pp. 6394–6398.

[75] J. Barker, S. Watanabe, E. Vincent, and J. Trmal. "The fifth CHiME Speech Separation and Recognition Challenge: Dataset, task and baselines". In: *arXiv preprint:1803.10609* (2018).

[76] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever. "Robust speech recognition via large-scale weak supervision". In: *arXiv preprint arXiv:2212.04356* (2022).

[77] J. G. Fiscus. "A post-processing system to yield reduced word error rates: Recognizer Output Voting Error Reduction (ROVER)". In: *Proc. of ASRU.* 1997.

[78] M. Wolf and C. Nadeu. "Towards microphone selection based on room impulse response energy-related measures". In: *Proc. of I Joint SIG-IL/Microsoft Workshop on Speech and Language Technologies for Iberian Languages, Porto Salvo, Portugal.* 2009, pp. 61–64.

[79] M. Wolf and C. Nadeu Camprubi. "On the potential of channel selection for recognition of reverberated speech with multiple microphones". In: *Proc. of Interspeech).* 2010, pp. 574–577.

[80] K. Kumatani, J. McDonough, J. F. Lehman, and B. Raj. "Channel selection based on multichannel cross-correlation coefficients for distant speech recognition". In: *2011 Joint Workshop on Hands-free Speech Communication and Microphone Arrays.* IEEE. 2011, pp. 1–6.

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

[82] C. Guerrero Flores, G. Tryfou, and M. Omologo. “Cepstral Distance Based Channel Selection for Distant Speech Recognition”. In: *Computer Speech and Languages* (2018), pp. 314–332.

[83] M. Wölfel. “Channel selection by class separability measures for automatic transcriptions on distant microphones”. In: *Proc. of Interspeech.* 2007.

[84] Y. Obuchi. “Multiple-microphone robust speech recognition using decoder-based channel selection”. In: *ISCA Tutorial and Research Workshop (ITRW) on Statistical and Perceptual Audio Processing.* 2004.

[85] Y. Obuchi. “Noise robust speech recognition using delta-cepstrum normalization and channel selection”. In: *Electronics and Communications in Japan (Part II: Electronics)* 89.7 (2006), pp. 9–20.

[86] M. Wolf and C. Nadeu. “Pairwise likelihood normalization-based channel selection for multi-microphone ASR”. In: *Proc. of Iber-SPEECH,(Madrid, Spain)* (2012), pp. 513–522.

[87] F. Xiong, J. Zhang, B. Meyer, H. Christensen, and J. Barker. “Channel selection using neural network posterior probability for speech recognition with distributed microphone arrays in everyday environments”. In: *5th CHiME International Workshop on Speech Processing in Everyday Environments.* 2018, pp. 19–24.

[88] J. Chen and X.-L. Zhang. “Scaling sparsemax based channel selection for speech recognition with ad-hoc microphone arrays”. In: *Proc. of Interspeech* (2021).

[89] J. Barber, Y. Fan, and T. Zhang. “End-to-end Alexa device arbitration”. In: *Proc. of ICASSP.* IEEE. 2022, pp. 926–930.

[90] A. Martins and R. Astudillo. “From softmax to sparsemax: A sparse model of attention and multi-label classification”. In: *International conference on machine learning.* PMLR. 2016, pp. 1614–1623.

[91] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. “Learning to Rank Using Gradient Descent”. In: *International Conference on Machine Learning.* 2005, pp. 89–96.

[92] Z. Cap, T. Qin, T. Y. Liu, M. F. Tsai, and H. Li. “Learning to Rank: From Pairwise Approach to Listwise Approach”. In: *International Conference on Machine Learning.* 2007, pp. 129–136. ISBN: 9781595937933.

[93] N. Furnon, R. Serizel, I. Illina, and S. Essid. “DNN-based mask estimation for distributed speech enhancement in spatially unconstrained microphone arrays”. In: *submitted to TASLP.* 2020.

[94] D. Diaz-Guerra, A. Miguel, and J. R. Beltran. "gpuRIR: A python library for room impulse response simulation with GPU acceleration". In: *Multimedia Tools and Applications* 80.4 (2021), pp. 5653–5671.

[95] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, et al. "The Kaldi speech recognition toolkit". In: *Proc. of ASRU*. IEEE. 2011.

[96] V. Manohar, S. Chen, Z. Wang, Y. Fujita, S. Watanabe, and S. Khudanpur. "Acoustic Modeling for Overlapping Speech Recognition: JHU CHiME-5 Challenge System". In: *Proc. of ICASSP*. 2019, pp. 6665–6669.

[97] M. Van Segbroeck, A. Zaid, K. Kutsenko, C. Huerta, T. Nguyen, X. Luo, B. Hoffmeister, J. Trmal, M. Omologo, and R. Maas. "DiPCo-Dinner Party Corpus". In: *Proc. of Interspeech* (2020).

[98] L. Brandschain, D. Graff, C. Cieri, K. Walker, C. Caruso, and A. Neely. "Mixer 6". In: *Proc. of LREC*. 2010.

[99] S. Cornell, M. Omologo, S. Squartini, and E. Vincent. "Detecting and counting overlapping speakers in distant speech scenarios". In: *Proc. of Interspeech*. 2020, pp. 3107–3111.

[100] S. Cornell, T. Balestri, and T. Sénéchal. "Implicit acoustic echo cancellation for keyword spotting and device-directed speech detection". In: *Spoken Language Technology Workshop*. 2023, pp. 1052–1058.

[101] R. Takeda, K. Nakadai, T. Takahashi, K. Komatani, T. Ogata, and H. G. Okuno. "ICA-based efficient blind dereverberation and echo cancellation method for barge-in-able robot audition". In: *Proc. Proc. of ICASSP*. 2009, pp. 3677–3680.

[102] R. Takeda, K. Nakadai, T. Takahashi, K. Komatani, T. Ogata, and H. G. Okuno. "Efficient blind dereverberation and echo cancellation based on independent component analysis for actual acoustic signals". In: *Neural computation* 24.1 (2012), pp. 234–272.

[103] I. Nishimuta, K. Yoshii, K. Itoyama, and H. G. Okuno. "Development of a robot quizmaster with auditory functions for speech-based multiparty interaction". In: *SICE International Symposium on System Integration*. 2014, pp. 328–333.

[104] S. Ding, Y. Jia, K. Hu, and Q. Wang. "Textual Echo Cancellation". In: *arXiv preprint arXiv:2008.06006* (2020).

[105] N. Howard, A. Park, T. Z. Shabestary, A. Gruenstein, and R. Prabhavalkar. "A Neural Acoustic Echo Canceller Optimized Using An Automatic Speech Recognizer and Large Scale Synthetic Data". In: *Proc. Proc. of ICASSP*. 2021, pp. 7128–7132.

[106] S. H. Mallidi, R. Maas, K. Goehner, A. Rastrow, S. Matsoukas, and B. Hoffmeister. “Device-directed Utterance Detection”. In: *Proc. Proc. of Interspeech* (2018), pp. 1225–1228.

[107] C. Huang, R. Maas, S. H. Mallidi, and B. Hoffmeister. “A Study for Improving Device-Directed Speech Detection Toward Frictionless Human-Machine Interaction”. In: *Proc. Proc. of Interspeech* (2019), pp. 3342–3346.

[108] K. Gillespie, I. C. Konstantakopoulos, X. Guo, V. T. Vasudevan, and A. Sethy. “Improving Device Directedness Classification of Utterances With Semantic Lexical Features”. In: *Proc. Proc. of ICASSP*. 2020, pp. 7859–7863.

[109] A. Norouzian, B. Mazoure, D. Connolly, and D. Willett. “Exploring Attention Mechanism for Acoustic-based Classification of Speech Utterances into System-directed and Non-system-directed”. In: *Proc. Proc. of ICASSP*. 2019, pp. 7310–7314.

[110] J. Benesty, T. Gänsler, D. R. Morgan, M. M. Sondhi, S. L. Gay, et al. *Advances in network and acoustic echo cancellation*. Springer, 2001.

[111] E. Hänsler and G. Schmidt. *Acoustic echo and noise control: a practical approach*. Vol. 40. Wiley-Interscience, Hoboken, 2005.

[112] D. A. Bendersky, J. W. Stokes, and H. S. Malvar. “Nonlinear residual acoustic echo suppression for high levels of harmonic distortion”. In: *Proc. Proc. of ICASSP*. 2008, pp. 261–264.

[113] A. Schwarz, C. Hofmann, and W. Kellermann. “Spectral feature-based nonlinear residual echo suppression”. In: *Proc. Proc. of ICASSP*. 2013, pp. 1–4.

[114] B. Panda, A. Kar, and M. Chandra. “Non-linear adaptive echo supression algorithms: A technical survey”. In: *Proc. Proc. of ICASSP*. 2014, pp. 076–080.

[115] H. Zhang and D. Wang. “Deep learning for acoustic echo cancellation in noisy and double-talk scenarios”. In: *Training* 161.2 (2018), p. 322.

[116] Q. Lei, H. Chen, J. Hou, L. Chen, and L. Dai. “Deep neural network based regression approach for acoustic echo cancellation”. In: *Proc. ICMSSP*. 2019, pp. 94–98.

[117] A. Fazel, M. El-Khamy, and J. Lee. “Deep Multitask Acoustic Echo Cancellation.” In: *Proc. Proc. of Interspeech*. 2019, pp. 4250–4254.

[118] H. Zhang, K. Tan, and D. Wang. “Deep Learning for Joint Acoustic Echo and Noise Cancellation with Nonlinear Distortions.” In: *Proc. Proc. of Interspeech*. 2019, pp. 4255–4259.

[119] A. Fazel, M. El-Khamy, and J. Lee. “Cad-aec: Context-aware deep acoustic echo cancellation”. In: *Proc. Proc. of ICASSP*. 2020, pp. 6919–6923.

[120] S. Ioffe and C. Szegedy. “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift”. In: *International Conference on Machine Learning*. 2015, pp. 448–456.

[121] P. Warden. “Speech Commands: A Dataset for Limited-Vocabulary Speech Recognition”. In: *ArXiv e-prints* (2018). arXiv: `1804.03209 [cs.CL]`.

[122] H. Zen, R. Clark, R. J. Weiss, V. Dang, Y. Jia, Y. Wu, Y. Zhang, and Z. Chen. “LibriTTS: A Corpus Derived from LibriSpeech for Text-to-Speech”. In: *Proc. Proc. of Interspeech*. 2019.

[123] D. Snyder, G. Chen, and D. Povey. “MUSAN: A music, speech, and noise corpus”. In: *arXiv preprint arXiv:1510.08484* (2015).

[124] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le. “SpecAugment: A Simple Augmentation Method for Automatic Speech Recognition”. In: *Proc. Proc. of Interspeech*. 2019.

[125] R. Scheibler, E. Bezzam, and I. Dokmanić. “Pyroomacoustics: A python package for audio room simulation and array processing algorithms”. In: *Proc. Proc. of ICASSP*. IEEE. 2018, pp. 351–355.

[126] S. Majumdar and B. Ginsburg. “MatchboxNet: 1D Time-Channel Separable Convolutional Neural Network Architecture for Speech Commands Recognition”. In: *Proc. Proc. of Interspeech* (2020).

[127] O. Kuchaiev, J. Li, H. Nguyen, O. Hrinchuk, R. Leary, B. Ginsburg, S. Kriman, S. Beliaev, V. Lavrukhin, J. Cook, et al. “Nemo: a toolkit for building ai applications using neural modules”. In: *arXiv preprint arXiv:1909.09577* (2019).

[128] S. Cornell, M. Omologo, S. Squartini, and E. Vincent. “Overlapped speech detection and speaker counting using distant microphone arrays”. In: *Computer Speech & Language* 72 (2022), p. 101306.

[129] L. R. Rabiner and M. R. Sambur. “An algorithm for determining the endpoints of isolated utterances”. In: *Bell System Technical Journal* 54.2 (1975), pp. 297–315.

[130] R. Tucker. “Voice activity detection using a periodicity measure”. In: *IEE Proceedings I (Communications, Speech and Vision)* 139.4 (1992), pp. 377–380.

[131] L. Rabiner and M. Sambur. “Application of an LPC distance measure to the voiced-unvoiced-silence detection problem”. In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 25.4 (1977), pp. 338–343.

[132] D. Freeman, G. Cosier, C. Southcott, and I. Boyd. “The voice activity detector for the Pan-European digital cellular mobile telephone service”. In: *Proc. of ICASSP*. IEEE. 1989, pp. 369–372.

[133] J. Ramirez, J. C. Segura, C. Benitez, A. d. l. Torre, and A. J. Rubio. “A new adaptive long-term spectral estimation voice activity detector”. In: *Eighth European Conference on Speech Communication and Technology*. 2003.

[134] J. Sohn, N. S. Kim, and W. Sung. “A statistical model-based voice activity detection”. In: *IEEE signal processing letters* 6.1 (1999), pp. 1–3.

[135] A. Benyassine, E. Shlomot, H. Su, D. Massaloux, C. Lamblin, and J. Petit. “A silence compression scheme for g. 729 optimized for terminals conforming to recommendation v. 70”. In: *Communications Magazine, IEEE* 35 (1997), pp. 64–73.

[136] J. W. Shin, J.-H. Chang, H. S. Yun, and N. S. Kim. “Voice activity detection based on generalized gamma distribution”. In: *Proc. of ICASSP*. Vol. 1. IEEE. 2005, pp. I–781.

[137] J.-H. Chang, N. S. Kim, and S. K. Mitra. “Voice activity detection based on multiple statistical models”. In: *IEEE Transactions on Signal Processing* 54.6 (2006), pp. 1965–1976.

[138] D. Enqing, L. Guizhong, Z. Yatong, and Z. Xiaodi. “Applying support vector machines to voice activity detection”. In: *Proc. of ICASSP*. Vol. 2. IEEE. 2002, pp. 1124–1127.

[139] J. Ramırez, P. Yélamos, J. Górriz, and J. Segura. “SVM-based speech endpoint detection using contextual speech features”. In: *Electronics letters* 42.7 (2006), pp. 877–879.

[140] M. Zelenák. “Detection and handling of overlapping speech for speaker diarization”. PhD thesis. Universitat Politècnica de Catalunya, 2012.

[141] S. N. Wrigley, G. J. Brown, V. Wan, and S. Renals. “Speech and crosstalk detection in multichannel audio”. In: *IEEE Transactions on speech and audio processing* 13.1 (2004), pp. 84–91.

[142] F. Eyben, F. Weninger, S. Squartini, and B. Schuller. “Real-life voice activity detection with lstm recurrent neural networks and an application to hollywood movies”. In: *Proc. of ICASSP*. IEEE. 2013, pp. 483–487.

[143] J. H. Ko, J. Fromm, M. Philipose, I. Tashev, and S. Zarar. “Limiting numerical precision of neural networks to achieve real-time voice activity detection”. In: *Proc. of ICASSP*. IEEE. 2018, pp. 2236–2240.

[144] R. Zazo, T. N. Sainath, G. Simko, and C. Parada. “Feature Learning with Raw-Waveform CLDNNs for Voice Activity Detection”. In: *Proc. of Interspeech* (2016), pp. 3668–3672.

[145] F. Vesperini, P. Vecchiotti, E. Principi, S. Squartini, and F. Piazza. “A neural network based algorithm for speaker localization in a multi-room environment”. In: *Proc. of MLSP*. IEEE. 2016, pp. 1–6.

[146] P. Vecchiotti, E. Principi, S. Squartini, and F. Piazza. “Deep neural networks for joint voice activity detection and speaker localization”. In: *Proc. of EUSIPCO*. IEEE. 2018, pp. 1567–1571.

[147] H. Bredin, R. Yin, J. M. Coria, G. Gelly, P. Korshunov, M. Lavechin, D. Fustes, H. Titeux, W. Bouaziz, and M.-P. Gill. “Pyannote. audio: neural building blocks for speaker diarization”. In: *Proc. of ICASSP*. IEEE. 2020, pp. 7124–7128.

[148] M. Ravanelli and Y. Bengio. “Speaker recognition from raw waveform with sincnet”. In: *Proc. of SLT*. IEEE. 2018, pp. 1021–1028.

[149] K. Boakye, O. Vinyals, and G. Friedland. “Improved overlapped speech handling for speaker diarization”. In: *Proc. of Interspeech*. 2011, pp. 941–944.

[150] R. Vipperla, J. T. Geiger, S. Bozonnet, D. Wang, N. Evans, B. Schuller, and G. Rigoll. “Speech overlap detection and attribution using convolutive non-negative sparse coding”. In: *Proc. of ICASSP*. 2012, pp. 4181–4184.

[151] D. Charlet, C. Barras, and J.-S. Liénard. “Impact of overlapping speech detection on speaker diarization for broadcast news and debates”. In: *Proc. of ICASSP*. 2013, pp. 7707–7711.

[152] S. H. Yella and H. Bourlard. “Overlapping speech detection using long-term conversational features for speaker diarization in meeting room conversations”. In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 22.12 (2014), pp. 1688–1700.

[153] S. Lee, J. Kim, J. Park, and M. Hahn. “Overlapping Speech Detection with Cluster-based HMM Framework”. In: *8th International Conference on Signal Processing Systems*. 2016, pp. 138–141.

[154] J. Geiger, F. Eyben, B. Schuller, and G. Rigoll. “Detecting overlapping speech with long short-term memory recurrent neural networks”. In: *Proc. of Interspeech*. 2013, pp. 1668–1672.

[155] J. Carletta, S. Ashby, S. Bourban, M. Flynn, M. Guillemot, T. Hain, J. Kadlec, V. Karaiskos, W. Kraaij, M. Kronenthal, et al. “The AMI meeting corpus: A pre-announcement”. In: *International workshop on machine learning for multimodal interaction*. Springer. 2005, pp. 28–39.

[156] L. Drude, A. Chinaev, D. H. T. Vu, and R. Haeb-Umbach. “Source counting in speech mixtures using a variational EM approach for complex Watson mixture models”. In: *Proc. of ICASSP*. 2014, pp. 6834–6838.

[157] S. Pasha, J. Donley, and C. Ritz. “Blind speaker counting in highly reverberant environments by clustering coherence features”. In: *2017 APSIPA Annual Summit and Conference*. 2017, pp. 1684–1687.

[158] A. Brutti, M. Omologo, and P. Svaizer. “Multiple Source Localization Based on Acoustic Map De-Emphasis”. In: *EURASIP Journal on Audio, Speech, and Music Processing* 2010.1 (2010), pp. 1–17.

[159] D. Pavlidi, A. Griffin, M. Puigt, and A. Mouchtaris. “Source counting in real-time sound source localization using a circular microphone array”. In: *2012 IEEE Sensor Array and Multichannel Signal Processing Workshop (SAM)*. 2012, pp. 521–524.

[160] T. Arai. “Estimating number of speakers by the modulation characteristics of speech”. In: *Proc. of ICASSP*. Vol. 2. 2003, pp. II–197.

[161] S. Ouamour, M. Guerti, and H. Sayoud. “PENS: a confidence parameter estimating the number of speakers”. In: *Second ISCA Workshop on Experimental Linguistics*. 2008, pp. 177–180.

[162] C. Xu, S. Li, et al. “Crowd++: unsupervised speaker count with smartphones”. In: *2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. 2013, pp. 43–52.

[163] V. Andrei, H. Cucu, A. Buzo, and C. Burileanu. “Counting Competing Speakers in a Timeframe — Human versus Computer”. In: *Proc. of Interspeech*. 2015, pp. 3399–3403.

[164] F. R. Stöter, S. Chakrabarty, B. Edler, and E. A. P. Habets. “CountNet: Estimating the Number of Concurrent Speakers Using Supervised Learning”. In: *IEEE/ACM Transactions on Audio, Speech and Language Processing* 27.2 (2019), pp. 268–282.

[165] V. Andrei, H. Cucu, and C. Burileanu. “Overlapped speech detection and competing speaker counting — humans versus deep learning”. In: *IEEE Journal of Selected Topics in Signal Processing* 13.4 (2019), pp. 850–862.

[166] N. Kanda, Y. Gaur, X. Wang, Z. Meng, Z. Chen, T. Zhou, and T. Yoshioka. "Joint Speaker Counting, Speech Recognition, and Speaker Identification for Overlapped Speech of any Number of Speakers". In: *Proc. of Interspeech.* 2020, pp. 36–40.

[167] V. Andrei, H. Cucu, and C. Burileanu. "Detecting Overlapped Speech on Short Timeframes Using Deep Learning". In: *Proc. of Interspeech.* 2017, pp. 1198–1202.

[168] N. Sajjan, S. Ganesh, N. Sharma, S. Ganapathy, and N. Ryant. "Leveraging LSTM models for overlap detection in multi-party meetings". In: *Proc. of ICASSP.* 2018, pp. 5249–5253.

[169] M. Kunešová, M. Hrúz, Z. Zajic, and V. Radová. "Detection of Overlapping Speech for the Purposes of Speaker Diarization". In: *International Conference on Speech and Computer.* 2019, pp. 247–257.

[170] L. Bullock, H. Bredin, and L. P. Garcia-Perera. "Overlap-aware diarization: Resegmentation using neural end-to-end overlapped speech detection". In: *Proc. of ICASSP.* IEEE. 2020, pp. 7114–7118.

[171] J. Málek and J. Žd'ánskỳ. "Voice-Activity and Overlapped Speech Detection Using x-Vectors". In: *International Conference on Text, Speech, and Dialogue.* 2020, pp. 366–376.

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

[173] J. L. Ba, J. R. Kiros, and G. E. Hinton. "Layer Normalization". In: *Stat* 1050 (2016), p. 21.

[174] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. "Dropout: a simple way to prevent neural networks from overfitting". In: *Journal of Machine Learning Research* 15.1 (2014), pp. 1929–1958.

[175] S. Bai, J. Z. Kolter, and V. Koltun. "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling". In: *arXiv preprint arXiv:1803.01271* (2018).

[176] Y. Luo and N. Mesgarani. "Conv-TasNet: Surpassing ideal time–frequency magnitude masking for speech separation". In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 27.8 (2019), pp. 1256–1266.

[177] 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". In: *arXiv preprint arXiv:1704.04861* (2017).

[178] K. He, X. Zhang, S. Ren, and J. Sun. “Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification”. In: *2015 IEEE International Conference on Computer Vision (ICCV)*. 2015, pp. 1026–1034.

[179] Y. Fujita, N. Kanda, S. Horiguchi, Y. Xue, K. Nagamatsu, and S. Watanabe. “End-to-end neural speaker diarization with self-attention”. In: *Proc. of ASRU*. 2019, pp. 296–303.

[180] T. Q. Nguyen and J. Salazar. “Transformers without tears: Improving the normalization of self-attention”. In: *arXiv preprint arXiv:1910.05895* (2019).

[181] C. Knapp and G. Carter. “The generalized correlation method for estimation of time delay”. In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 24.4 (1976), pp. 320–327.

[182] O. Walter, L. Drude, and R. Haeb-Umbach. “Source counting in speech mixtures by nonparametric Bayesian estimation of an infinite Gaussian mixture model”. In: *Proc. of ICASSP*. 2015, pp. 459–463.

[183] X. Xiao, S. Zhao, X. Zhong, D. L. Jones, E. S. Chng, and H. Li. “A learning-based approach to direction of arrival estimation in noisy and reverberant environments”. In: *Proc. of ICASSP*. 2015, pp. 2814–2818.

[184] S. Sivasankaran, E. Vincent, and D. Fohr. “Keyword-based speaker localization: Localizing a target speaker in a multi-speaker environment”. In: *Proc. of Interspeech*. 2020, pp. 2703–2707.

[185] S. Chakrabarty and E. A. P. Habets. “Broadband DOA estimation using convolutional neural networks trained with noise signals”. In: *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. 2017, pp. 136–140.

[186] S. Adavanne, A. Politis, and T. Virtanen. “Direction of arrival estimation for multiple sound sources using convolutional recurrent neural network”. In: *26th European Signal Processing Conference (EUSIPCO)*. 2018, pp. 1462–1466.

[187] P. Vecchiotti, N. Ma, S. Squartini, and G. J. Brown. “End-to-end binaural sound localisation from the raw waveform”. In: *Proc. of ICASSP*. 2019, pp. 451–455.

[188] S. Sivasankaran. “Localization guided speech separation”. PhD thesis. Université de Lorraine, Sept. 2020.

[189] E. Perez, F. Strub, H. De Vries, V. Dumoulin, and A. Courville. “Film: Visual reasoning with a general conditioning layer”. In: *32nd AAAI Conference on Artificial Intelligence*. 2018, pp. 3942–3951.

[190] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur. “Librispeech: an ASR corpus based on public domain audio books”. In: *Proc. of ICASSP*. 2015, pp. 5206–5210.

[191] M. McAuliffe, M. Socolof, S. Mihuc, M. Wagner, and M. Sonderegger. “Montreal Forced Aligner: Trainable Text-Speech Alignment Using Kaldi”. In: *Proc. of Interspeech*. 2017, pp. 498–502.

[192] D. Diaz-Guerra, A. Miguel, and J. R. Beltran. “gpuRIR: A Python library for Room Impulse Response simulation with GPU acceleration”. In: *arXiv preprint:1810.11359* (2018).

[193] N. Furnon, R. Serizel, I. Illina, and S. Essid. “Distributed speech separation in spatially unconstrained microphone arrays”. In: *arXiv preprint arXiv:2011.00982* (2020).

[194] I. McCowan, J. Carletta, W. Kraaij, S. Ashby, S. Bourban, M. Flynn, M. Guillemot, T. Hain, J. Kadlec, V. Karaiskos, M. Kronenthal, G. Lathoud, M. Lincoln, A. Lisowska Masson, W. Post, D. Reidsma, and P. Wellner. “The AMI meeting corpus”. In: *5th International Conference on Methods and Techniques in Behavioral Research*. 2005, pp. 137–140.

[195] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. “Microsoft COCO: Common objects in context”. In: *European Conference on Computer Vision (ECCV)*. 2014, pp. 740–755.

[196] K. Kishida. “Property of average precision and its generalization: An examination of evaluation indicator for information retrieval experiments”. In: *NII Technical Reports* 2005.14 (2005), pp. 1–19.

[197] J. Demšar. “Statistical comparisons of classifiers over multiple data sets”. In: *Journal of Machine Learning Research* 7.Jan (2006), pp. 1–30.

[198] L. Liu, H. Jiang, P. He, W. Chen, X. Liu, J. Gao, and J. Han. “On the Variance of the Adaptive Learning Rate and Beyond”. In: *International Conference on Learning Representations (ICLR)*. 2020.

[199] D. S. Park, W. Chan, Y. Zhang, C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le. “SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition”. In: *Proc. of Interspeech* (2019), pp. 2613–2617.

[200] D. Terpstra, H. Jagode, H. You, and J. Dongarra. “Collecting performance data with PAPI-C”. In: *3rd International Workshop on Parallel Tools for High Performance Computing*. 2009, pp. 157–173.

[201] S. Choi, S. Seo, B. Shin, H. Byun, M. Kersner, B. Kim, D. Kim, and S. Ha. “Temporal Convolution for Real-Time Keyword Spotting on Mobile Devices”. In: *Proc. of Interspeech*. 2019, pp. 3372–3376.

[202] S. Sivasankaran, E. Vincent, and D. Fohr. "Analyzing the impact of speaker localization errors on speech separation for automatic speech recognition". In: *28th European Signal Processing Conference (EUSIPCO).* 2021.

[203] G. Morrone, S. Cornell, D. Raj, L. Serafini, E. Zovato, A. Brutti, and S. Squartini. "Low-Latency Speech Separation Guided Diarization for Telephone Conversations". In: *Spoken Language Technology Workshop.* 2023, pp. 641–646.

[204] L. Serafini, S. Cornell, G. Morrone, E. Zovato, A. Brutti, and S. Squartini. "An Experimental Review of Speaker Diarization methods with application to Two-Speaker Conversational Telephone Speech recordings". In: *Submitted to Speech Communication* (2022).

[205] H. Gish, M.-H. Siu, and J. R. Rohlicek. "Segregation of speakers for speech recognition and speaker identification". In: *Proc. of ICASSP.* 1991, pp. 873–876.

[206] M.-H. Siu, G. Yu, and H. Gish. "An unsupervised, sequential learning algorithm for the segmentation of speech waveforms with multiple speakers". In: *Proc. of ICASSP.* IEEE. 1992, pp. 189–192.

[207] U. Jain, M. A. Siegler, S.-J. Doh, E. Gouvea, J. Huerta, P. J. Moreno, B. Raj, and R. M. Stern. "Recognition of continuous broadcast news with multiple unknown speakers and environments". In: *Proc. of DARPA Speech Recognition Workshop.* 1996, pp. 61–66.

[208] S. Chen, P. Gopalakrishnan, et al. "Speaker, environment and channel change detection and clustering via the bayesian information criterion". In: *Proc. of DARPA broadcast news transcription and understanding workshop.* 1998, pp. 127–132.

[209] D. Liu and F. Kubala. "Fast speaker change detection for broadcast news transcription and indexing". In: *Proc. of Eurospeech.* 1999, pp. 1031–1034.

[210] H. Hermansky. "Perceptual linear predictive (PLP) analysis of speech". In: *The Journal of the Acoustical Society of America* 87.4 (1990), pp. 1738–1752.

[211] D. O'Shaughnessy. "Linear predictive coding". In: *IEEE Potentials* 7.1 (1988), pp. 29–32.

[212] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn. "Speaker verification using adapted Gaussian mixture models". In: *Digital Signal Processing* 10.1-3 (2000), pp. 19–41.

[213] P. Kenny, G. Boulianne, P. Ouellet, and P. Dumouchel. “Speaker and session variability in GMM-based speaker verification”. In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 15.4 (2007), pp. 1448–1460.

[214] F. Castaldo, D. Colibro, E. Dalmasso, P. Laface, and C. Vair. “Stream-based speaker segmentation using speaker factors and eigenvoices”. In: *Proc. of ICASSP*. IEEE. 2008, pp. 4133–4136.

[215] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet. “Front-end factor analysis for speaker verification”. In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 19.4 (2010), pp. 788–798.

[216] P. Matějka, O. Glembek, F. Castaldo, M. J. Alam, O. Plchot, P. Kenny, L. Burget, and J. Černocky. “Full-covariance UBM and heavy-tailed PLDA in i-vector speaker verification”. In: *Proc. of ICASSP*. IEEE. 2011, pp. 4828–4831.

[217] E. Variani, X. Lei, E. McDermott, I. L. Moreno, and J. Gonzalez-Dominguez. “Deep neural networks for small footprint text-dependent speaker verification”. In: *Proc. of ICASSP*. IEEE. 2014, pp. 4052–4056.

[218] Z. Bai and X.-L. Zhang. “Speaker recognition based on deep learning: An overview”. In: *Neural Networks* 140 (2021), pp. 65–99.

[219] F. Landini, O. Glembek, P. Matějka, J. Rohdin, L. Burget, M. Diez, and A. Silnova. “Analysis of the BUT diarization system for VoxConverse challenge”. In: *Proc. of ICASSP*. IEEE. 2021, pp. 5819–5823.

[220] N. R. Koluguri, T. Park, and B. Ginsburg. “TitaNet: Neural Model for speaker representation with 1D Depth-wise separable convolutions and global context”. In: *Proc. of ICASSP*. IEEE. 2022, pp. 8102–8106.

[221] B. Desplanques, J. Thienpondt, and K. Demuynck. “ECAPA-TDNN: Emphasized Channel Attention, Propagation and Aggregation in TDNN Based Speaker Verification”. In: *Proc. of Interspeech*. 2020, pp. 3830–3834.

[222] H. Bredin. “TristouNet: triplet loss for speaker turn embedding”. In: *Proc. of ICASSP*. IEEE. 2017, pp. 5430–5434.

[223] Y. Li, F. Gao, Z. Ou, and J. Sun. “Angular softmax loss for end-to-end speaker verification”. In: *Proc. of ISCSLP*. IEEE. 2018, pp. 190–194.

[224] J. S. Chung, J. Huh, S. Mun, M. Lee, H. S. Heo, S. Choe, C. Ham, S. Jung, B.-J. Lee, and I. Han. “In defence of metric learning for speaker recognition”. In: *Proc. of Interspeech*. 2020, pp. 2977–2981.

[225] T. J. Park, K. J. Han, M. Kumar, and S. Narayanan. “Auto-tuning spectral clustering for speaker diarization using normalized maximum eigengap”. In: *IEEE SPS* 27 (2019), pp. 381–385.

[226] M. Diez, L. Burget, and P. Matejka. “Speaker Diarization based on Bayesian HMM with Eigenvoice Priors”. In: *Proc. of Odyssey*. 2018, pp. 147–154.

[227] F. Landini, J. Profant, M. Diez, and L. Burget. “Bayesian HMM clustering of x-vector sequences (VBx) in speaker diarization: theory, implementation and analysis on standard tasks”. In: *Computer Speech & Language* 71 (2020), p. 101254.

[228] F. Landini, A. Lozano-Diez, M. Diez, and L. Burget. “From Simulated Mixtures to Simulated Conversations as Training Data for End-to-End Neural Diarization”. In: *Proc. of Interspeech* (2022), pp. 5095–5099.

[229] N. Ryant, K. Church, C. Cieri, A. Cristia, J. Du, S. Ganapathy, and M. Liberman. “Second DIHARD challenge evaluation plan”. In: *Linguistic Data Consortium, Tech. Rep* (2019).

[230] A. Stolcke and T. Yoshioka. “DOVER: A method for combining diarization outputs”. In: *Proc. of ASRU*. IEEE. 2019, pp. 757–763.

[231] D. Raj, L. P. Garcia-Perera, Z. Huang, S. Watanabe, D. Povey, A. Stolcke, and S. Khudanpur. “DOVER-Lap: A method for combining overlap-aware diarization outputs”. In: *Proc. of SLT*. IEEE. 2021, pp. 881–888.

[232] D. Raj and S. Khudanpur. “Reformulating DOVER-Lap Label Mapping as a Graph Partitioning Problem”. In: *Proc. of Interspeech*. 2021, pp. 2351–2355.

[233] D. Raj, Z. Huang, and S. Khudanpur. “Multi-class spectral clustering with overlaps for speaker diarization”. In: *Proc. of SLT*. IEEE. 2021, pp. 582–589.

[234] Z. Huang, S. Watanabe, Y. Fujita, P. Garcia, Y. Shao, D. Povey, and S. Khudanpur. “Speaker diarization with region proposal network”. In: *Proc. of ICASSP*. IEEE. 2020, pp. 6514–6518.

[235] A. Zhang, Q. Wang, Z. Zhu, J. Paisley, and C. Wang. “Fully supervised speaker diarization”. In: *Proc. of ICASSP*. IEEE. 2019, pp. 6301–6305.

[236] Q. Li, F. L. Kreyssig, C. Zhang, and P. C. Woodland. “Discriminative neural clustering for speaker diarisation”. In: *Proc. of SLT*. IEEE. 2021, pp. 574–581.

[237] X. Fang, Z.-H. Ling, L. Sun, S.-T. Niu, J. Du, C. Liu, and Z.-C. Sheng. “A deep analysis of speech separation guided diarization under realistic conditions”. In: *Proc. of APSIPA ASC*. IEEE. 2021, pp. 667–671.

[238] I. Medennikov, M. Korenevsky, T. Prisyach, Y. Khokhlov, M. Korenevskaya, I. Sorokin, T. Timofeeva, A. Mitrofanov, A. Andrusenko, I. Podluzhny, et al. "Target-Speaker Voice Activity Detection: A Novel Approach for Multi-Speaker Diarization in a Dinner Party Scenario". In: *Proc. of Interspeech* (2020), pp. 274–278.

[239] N. Ryant, K. Church, C. Cieri, J. Du, S. Ganapathy, and M. Liberman. "Third DIHARD challenge evaluation plan". In: *Proc. of Interspeech*. 2021, pp. 3570–3574.

[240] Y. Wang, M. He, S. Niu, L. Sun, T. Gao, X. Fang, J. Pan, J. Du, and C.-H. Lee. "USTC-NELSLIP system description for DIHARD-III challenge". In: *Proc. of 3rd DIHARD Speech Diarization Challenge Workshop*. 2021.

[241] M. Kolbæk, D. Yu, Z.-H. Tan, and J. Jensen. "Multitalker speech separation with utterance-level permutation invariant training of deep recurrent neural networks". In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 25.10 (2017), pp. 1901–1913.

[242] Y. Fujita, N. Kanda, S. Horiguchi, K. Nagamatsu, and S. Watanabe. "End-to-End Neural Speaker Diarization with Permutation-free Objectives". In: *Proc. of Interspeech*. 2019, pp. 4300–4304.

[243] Y. Fujita, N. Kanda, S. Horiguchi, Y. Xue, K. Nagamatsu, and S. Watanabe. "End-to-end neural speaker diarization with self-attention". In: *Proc. of ASRU*. IEEE. 2019, pp. 296–303.

[244] S. Horiguchi, Y. Fujita, S. Watanabe, Y. Xue, and K. Nagamatsu. "End-to-End Speaker Diarization for an Unknown Number of Speakers with Encoder-Decoder Based Attractors". In: *Proc. of Interspeech*. 2020, pp. 269–273.

[245] K. Kinoshita, M. Delcroix, and N. Tawara. "Integrating end-to-end neural and clustering-based diarization: Getting the best of both worlds". In: *Proc. of ICASSP*. IEEE. 2021, pp. 7198–7202.

[246] K. Kinoshita, M. Delcroix, and N. Tawara. "Advances in integration of end-to-end neural and clustering-based diarization for real conversational speech". In: *Proc. of Interspeech*. 2021, pp. 3565–3569.

[247] K. Kinoshita, T. von Neumann, M. Delcroix, C. Boeddeker, and R. Haeb-Umbach. "Utterance-by-utterance overlap-aware neural diarization with Graph-PIT". In: *Proc. of Interspeech*. 2022, pp. 1486–1490.

[248] K. Kinoshita, M. Delcroix, and T. Iwata. "Tight integration of neural- and clustering-based diarization through deep unfolding of infinite gaussian mixture model". In: *Proc. of ICASSP*. IEEE. 2022, pp. 8382–8386.

[249] Y. Xue, S. Horiguchi, Y. Fujita, S. Watanabe, P. Garcia, and K. Nagamatsu. "Online end-to-end neural diarization with speaker-tracing buffer". In: *Proc. of SLT*. IEEE. 2021, pp. 841–848.

[250] E. Han, C. Lee, and A. Stolcke. "BW-EDA-EEND: Streaming end-to-end neural speaker diarization for a variable number of speakers". In: *Proc. of ICASSP*. IEEE. 2021, pp. 7193–7197.

[251] S. Horiguchi, S. Watanabe, P. Garcia, Y. Takashima, and Y. Kawaguchi. "Online Neural Diarization of Unlimited Numbers of Speakers". In: *arXiv preprint arXiv:2206.02432* (2022).

[252] J. M. Coria, H. Bredin, S. Ghannay, and S. Rosset. "Overlap-aware Low-Latency Online Speaker Diarization based on End-to-End Local Segmentation". In: *Proc. of ASRU*. IEEE. 2021, pp. 1139–1146.

[253] S. Horiguchi, N. Yalta, P. Garcia, Y. Takashima, Y. Xue, D. Raj, Z. Huang, Y. Fujita, S. Watanabe, and S. Khudanpur. "The Hitachi-JHU DIHARD III system: Competitive end-to-end neural diarization and x-vector clustering systems combined by DOVER-Lap". In: *3rd DIHARD Speech Diarization Challenge Workshop*. 2021.

[254] F. Landini, A. Lozano-Diez, L. Burget, M. Diez, A. Silnova, K. Zmolıková, O. Glembek, P. Matejka, T. Stafylakis, and N. Brümmer. "BUT system description for the third DIHARD speech diarization challenge". In: *Proc. of 3rd DIHARD Speech Diarization Challenge Workshop*. 2021.

[255] N. Zeghidour, O. Teboul, and D. Grangier. "DIVE: End-to-end speech diarization via iterative speaker embedding". In: *Proc. of ASRU*. IEEE. 2021, pp. 702–709.

[256] L. E. Shafey, H. Soltau, and I. Shafran. "Joint speech recognition and speaker diarization via sequence transduction". In: *arXiv preprint arXiv:1907.05337* (2019).

[257] Z. Chen, T. Yoshioka, L. Lu, T. Zhou, Z. Meng, Y. Luo, J. Wu, X. Xiao, and J. Li. "Continuous speech separation: Dataset and analysis". In: *Proc. of ICASSP*. IEEE. 2020, pp. 7284–7288.

[258] D. Raj, P. Denisov, Z. Chen, H. Erdogan, Z. Huang, M. He, S. Watanabe, J. Du, T. Yoshioka, Y. Luo, et al. "Integration of speech separation, diarization, and recognition for multi-speaker meetings: System description, comparison, and analysis". In: *Proc. of SLT*. IEEE. 2021, pp. 897–904.

[259] X. Xiao, N. Kanda, Z. Chen, T. Zhou, T. Yoshioka, et al. "Microsoft Speaker Diarization System for the VoxCeleb Speaker Recognition Challenge 2020". In: *Proc. of ICASSP*. IEEE. 2021, pp. 5824–5828.

[260] J. Le Roux, S. Wisdom, H. Erdogan, and J. R. Hershey. “SDR–half-baked or well done?” In: *Proc. of ICASSP*. IEEE. 2019, pp. 626–630.

[261] O. S. Language and Resources. *SRE Data. Available at* `https://openslr.org/10/` (accessed September 14, 2022).

[262] M. Pariente, S. Cornell, J. Cosentino, S. Sivasankaran, E. Tzinis, J. Heitkaemper, M. Olvera, F.-R. Stöter, M. Hu, J. M. Martín-Doñas, et al. “Asteroid: the PyTorch-based audio source separation toolkit for researchers”. In: *Proc. of Interspeech.* 2020.

[263] D. P. Kingma and J. Ba. “Adam: A Method for Stochastic Optimization”. In: *Proc. of ICLR.* 2015.

[264] NIST. *md-eval.pl (Version 22) in SCTK (version 2.4.12). Available at* `https://github.com/usnistgov/SCTK` ((accessed September 14, 2022).

[265] Y. Xue et al. “Online Streaming End-to-End Neural Diarization Handling Overlapping Speech and Flexible Numbers of Speakers”. In: *Proc. of Interspeech.* 2021, pp. 3116–3120.

[266] F. Landini, J. Profant, M. Diez, and L. Burget. “Bayesian HMM clustering of x-vector sequences (VBx) in speaker diarization: theory, implementation and analysis on standard tasks”. In: *Computer Speech & Language* 71 (2022), p. 101254.

[267] V. Peddinti, G. Chen, V. Manohar, T. Ko, D. Povey, and S. Khudanpur. “JHU ASpIRE system: Robust LVCSR with TDNNs, iVector adaptation and RNN-LMS”. In: *Proc. of ASRU.* IEEE. 2015, pp. 539–546.

[268] L. F. W. Anthony, B. Kanding, and R. Selvan. “Carbontracker: Tracking and predicting the carbon footprint of training deep learning models”. In: *ICML Workshop on Challenges in Deploying and monitoring Machine Learning Systems.* 2020.

[269] Y. Luo and N. Mesgarani. “TasNet: time-domain audio separation network for real-time, single-channel speech separation”. In: *CoRR* abs/1711.00541 (2017). arXiv: `1711.00541`. URL: `http://arxiv.org/abs/1711.00541`.

[270] J. Heymann, L. Drude, C. Boeddeker, P. Hanebrink, and R. B. Haeb-Umbach. “End-to-end training of a beamformer-supported multichannel ASR system”. In: *Proc. of ICASSP.* 2017, pp. 5–9.

[271] T. Nakatani, N. Ito, T. Higuchi, S. Araki, and K. Kinoshita. “Integrating DNN-based and spatial clustering-based mask estimation for robust MVDR beamforming”. In: *Proc. of ICASSP.* IEEE. 2017, pp. 286–290.

[272] C. Li, J. Shi, W. Zhang, A. S. Subramanian, X. Chang, N. Kamo, M. Hira, T. Hayashi, C. Boeddeker, Z. Chen, et al. “ESPnet-SE: End-to-end speech enhancement and separation toolkit designed for ASR integration”. In: *Proc. of SLT*. IEEE. 2021, pp. 785–792.

[273] W. Zhang, C. Boeddeker, S. Watanabe, T. Nakatani, M. Delcroix, K. Kinoshita, T. Ochiai, N. Kamo, R. Haeb-Umbach, and Y. Qian. “End-to-end dereverberation, beamforming, and speech recognition with improved numerical stability and advanced frontend”. In: *Proc. of ICASSP*. IEEE. 2021, pp. 6898–6902.

[274] X. Chang, T. Maekaku, Y. Fujita, and S. Watanabe. “End-to-end integration of speech recognition, speech enhancement, and self-supervised learning representation”. In: *Proc. of ICASSP* (2022).

[275] N. Kanda, X. Xiao, Y. Gaur, X. Wang, Z. Meng, Z. Chen, and T. Yoshioka. “Transcribe-to-diarize: Neural speaker diarization for unlimited number of speakers using end-to-end speaker-attributed asr”. In: *Proc. of ICASSP*. IEEE. 2022, pp. 8082–8086.

[276] A. Khare, E. Han, Y. Yang, and A. Stolcke. “Asr-aware end-to-end neural diarization”. In: *Proc. of ICASSP*. IEEE. 2022, pp. 8092–8096.

[277] S. Maiti, Y. Ueda, S. Watanabe, C. Zhang, M. Yu, S.-X. Zhang, and Y. Xu. “EEND-SS: Joint end-to-end neural speaker diarization and speech separation for flexible number of speakers”. In: *Proc. of SLT*. IEEE. 2023, pp. 480–487.

[278] H. Erdogan, J. R. Hershey, S. Watanabe, and J. Le Roux. “Phase-sensitive and recognition-boosted speech separation using deep recurrent neural networks”. In: *Proc. of ICASSP*. 2015.

[279] T. Ochiai, S. Watanabe, T. Hori, J. R. Hershey, and X. Xiao. “Unified architecture for multichannel end-to-end speech recognition with neural beamforming”. In: *IEEE Journal of Selected Topics in Signal Processing* 11.8 (2017), pp. 1274–1288.

[280] R. Gong, C. Quillen, D. Sharma, A. Goderre, J. Lainez, and L. Milanović. “Self-attention channel combinator frontend for end-to-end multichannel far-field speech recognition”. In: *Proc. of Interspeech* (2021).

[281] S. Dalmia, D. Okhonko, M. Lewis, S. Edunov, S. Watanabe, F. Metze, L. Zettlemoyer, and A. Mohamed. “Legonn: Building modular encoder-decoder models”. In: *arXiv preprint arXiv:2206.03318* (2022).

[282] T. Nakatani and K. Kinoshita. “Maximum likelihood convolutional beamformer for simultaneous denoising and dereverberation”. In: *Proc. of EUSIPCO*. IEEE. 2019, pp. 1–5.

[283] M. Ravanelli, P. Brakel, M. Omologo, and Y. Bengio. “A network of deep neural networks for distant speech recognition”. In: *Proc. of ICASSP*. IEEE. 2017, pp. 4880–4884.

[284] J. R. Hershey, Z. Chen, J. Le Roux, and S. Watanabe. “Deep clustering: Discriminative embeddings for segmentation and separation”. In: *Proc. of ICASSP*. 2016, pp. 31–35.

[285] E. Tzinis, Z. Wang, X. Jiang, and P. Smaragdis. “Compute and memory efficient universal sound source separation”. In: *Journal of Signal Processing Systems* 94.2 (2022), pp. 245–259.

[286] D. Seo, H.-S. Oh, and Y. Jung. “Wav2kws: Transfer learning from speech representations for keyword spotting”. In: *IEEE Access* 9 (2021), pp. 80682–80691.

[287] E. Tzinis, Y. Adi, V. K. Ithapu, B. Xu, P. Smaragdis, and A. Kumar. “RemixIT: Continual self-training of speech enhancement models via bootstrapped remixing”. In: *IEEE Journal of Selected Topics in Signal Processing* 16.6 (2022), pp. 1329–1341.

[288] K. Saijo and T. Ogawa. “Self-Remixing: Unsupervised Speech Separation VIA Separation and Remixing”. In: *Proc. of ICASSP*. IEEE. 2023, pp. 1–5.

[289] Z. Zhao, R. Gu, D. Yang, J. Tian, and Y. Zou. “Speaker-Aware Mixture of Mixtures Training for Weakly Supervised Speaker Extraction”. In: *arXiv preprint arXiv:2204.07375* (2022).

[290] A. Sivaraman, S. Wisdom, H. Erdogan, and J. R. Hershey. “Adapting speech separation to real-world meetings using mixture invariant training”. In: *Proc. of ICASSP*. IEEE. 2022, pp. 686–690.

[291] B. Thomee, D. A. Shamma, G. Friedland, B. Elizalde, K. Ni, D. Poland, D. Borth, and L.-J. Li. “YFCC100M: The new data in multimedia research”. In: *Communications of the ACM* 59.2 (2016), pp. 64–73.

[292] C. Gao, Y. Gu, F. Caliva, and Y. Liu. “Self-supervised speech representation learning for keyword-spotting with light-weight transformers”. In: *Proc. of ICASSP*. IEEE. 2023, pp. 1–5.

[293] Y. Lee, K. JANG, J. Goo, Y. Jung, and H.-R. Kim. “FitHuBERT: Going Thinner and Deeper for Knowledge Distillation of Speech Self-Supervised Learning”. In: *Proc. of Interspeech*. ISCA. 2022, pp. 3588–3592.

[294] P. Swarup, D. Chakrabarty, A. Sapru, H. Tulsiani, H. Arsikere, and S. Garimella. “Knowledge Distillation and Data Selection for Semi-Supervised Learning in CTC Acoustic Models”. In: *arXiv preprint arXiv:2008.03923* (2020).

[295] X. Yang, Q. Li, and P. C. Woodland. “Knowledge distillation for neural transducers from large self-supervised pre-trained models”. In: *Proc. of ICASSP*. IEEE. 2022, pp. 8527–8531.

[296] A. Ramesh, M. Pavlov, G. Goh, S. Gray, C. Voss, A. Radford, M. Chen, and I. Sutskever. “Zero-shot text-to-image generation”. In: *International Conference on Machine Learning*. PMLR. 2021, pp. 8821–8831.

[297] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. “High-resolution image synthesis with latent diffusion models”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 10684–10695.

[298] Z. Borsos, R. Marinier, D. Vincent, E. Kharitonov, O. Pietquin, M. Sharifi, O. Teboul, D. Grangier, M. Tagliasacchi, and N. Zeghidour. “Audiolm: a language modeling approach to audio generation”. In: *arXiv preprint arXiv:2209.03143* (2022).

[299] H. Liu, Z. Chen, Y. Yuan, X. Mei, X. Liu, D. Mandic, W. Wang, and M. D. Plumbley. “Audioldm: Text-to-audio generation with latent diffusion models”. In: *arXiv preprint arXiv:2301.12503* (2023).

[300] F. Schneider, Z. Jin, and B. Schölkopf. “Mo\^ usai: Text-to-Music Generation with Long-Context Latent Diffusion”. In: *arXiv preprint arXiv:2301.11757* (2023).

[301] C. Wang, S. Chen, Y. Wu, Z. Zhang, L. Zhou, S. Liu, Z. Chen, Y. Liu, H. Wang, J. Li, et al. “Neural Codec Language Models are Zero-Shot Text to Speech Synthesizers”. In: *arXiv preprint arXiv:2301.02111* (2023).

[302] T. A. Nguyen, E. Kharitonov, J. Copet, Y. Adi, W.-N. Hsu, A. Elkahky, P. Tomasello, R. Algayres, B. Sagot, A. Mohamed, et al. “Generative spoken dialogue language modeling”. In: *Transactions of the Association for Computational Linguistics* 11 (2023), pp. 250–266.

[303] A. Défossez, J. Copet, G. Synnaeve, and Y. Adi. “High fidelity neural audio compression”. In: *arXiv preprint arXiv:2210.13438* (2022).

[304] R. Huang, M. Li, D. Yang, J. Shi, X. Chang, Z. Ye, Y. Wu, Z. Hong, J. Huang, J. Liu, et al. “AudioGPT: Understanding and Generating Speech, Music, Sound, and Talking Head”. In: *arXiv preprint arXiv:2304.12995* (2023).

[305] F. P. Such, A. Rawal, J. Lehman, K. Stanley, and J. Clune. “Generative teaching networks: Accelerating neural architecture search by learning to generate synthetic training data”. In: *International Conference on Machine Learning*. PMLR. 2020, pp. 9206–9216.

[306] R. He, S. Sun, X. Yu, C. Xue, W. Zhang, P. Torr, S. Bai, and X. Qi. "Is synthetic data from generative models ready for image recognition?" In: *arXiv preprint arXiv:2210.07574* (2022).

[307] C. Subakan, M. Ravanelli, S. Cornell, F. Grondin, and M. Bronzi. "On using transformers for speech-separation". In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing (just accepted)* (2023).

[308] M. Severini, E. Principi, S. Cornell, L. Gabrielli, and S. Squartini. "Who Cried When: Infant Cry Diarization with Dilated Fully-Convolutional Neural Networks". In: *International Joint Conference on Neural Networks (IJCNN).* 2020, pp. 1–8.

[309] S. Cornell, E. Principi, and S. Squartini. "A Novel Adversarial Training Scheme for Deep Neural Network based Speech Enhancement". In: *International Joint Conference on Neural Networks (IJCNN).* 2020, pp. 1–8.

[310] S. Cornell, M. Olvera, M. Pariente, G. Pepe, E. Principi, L. Gabrielli, and S. Squartini. "Domain-adversarial training and trainable parallel front-end for the dcase 2020 task 4 sound event detection challenge". In: *DCASE 2020-5th Workshop on Detection and Classification of Acoustic Scenes and Events.* 2020.

[311] S. Cornell, M. Olvera, M. Pariente, G. Pepe, E. Principi, L. Gabrielli, and S. Squartini. "Task-aware separation for the dcase 2020 task 4 sound event detection and separation challenge". In: *DCASE 2020-5th Workshop on Detection and Classification of Acoustic Scenes and Events.* 2020.

[312] C. Subakan, M. Ravanelli, S. Cornell, M. Bronzi, and J. Zhong. "Attention is All You Need in Speech Separation". In: *Proc. of ICASSP* (2020).

[313] C. Aironi, S. Cornell, E. Principi, and S. Squartini. "Graph-based representation of audio signals for sound event classification". In: *Proc. of EUSIPCO.* IEEE. 2021, pp. 566–570.

[314] G. Morrone, S. Cornell, E. Zovato, A. Brutti, and S. Squartini. "Conversational Speech Separation: an Evaluation Study for Streaming Applications". In: *Proceedings of the 152nd International Audio Engineering Convention.* 2022.

[315] F. Ronchini, R. Serizel, N. Turpault, and S. Cornell. "The impact of non-target events in synthetic soundscapes for sound event detection". In: *DCASE 2021-Detection and Classification of Acoustic Scenes and Events.* 2021.

[316] C. Aironi, S. Cornell, E. Principi, and S. Squartini. “Graph Node Embeddings for ontology-aware Sound Event Classification: an evaluation study”. In: *Proc. of EUSIPCO*. IEEE. 2022, pp. 414–418.

[317] C. Subakan, M. Ravanelli, S. Cornell, and F. Grondin. “REAL-M: Towards speech separation on real mixtures”. In: *Proc. of ICASSP*. 2022, pp. 6862–6866.

[318] F. Ronchini, S. Cornell, R. Serizel, N. Turpault, E. Fonseca, and D. P. Ellis. “Description and analysis of novelties introduced in DCASE Task 4 2022 on the baseline system”. In: *DCASE 2022-7th Workshop on Detection and Classification of Acoustic Scenes and Events* (2022).

[319] Z.-Q. Wang, S. Cornell, S. Choi, Y. Lee, B.-Y. Kim, and S. Watanabe. “Neural Speech Enhancement with Very Low Algorithmic Latency and Complexity via Integrated full-and sub-band Modeling”. In: *Proc. of ICASSP*. IEEE. 2023, pp. 1–5.

[320] R. Serizel, S. Cornell, and N. Turpault. “Performance Above All? Energy Consumption vs. Performance, a Study on Sound Event Detection with Heterogeneous Data”. In: *Proc. of ICASSP*. IEEE. 2023, pp. 1–5.

[321] C. Aironi, S. Cornell, and S. Squartini. “Tackling the Linear Sum Assignment Problem with Graph Neural Networks”. In: *Applied Intelligence and Informatics: Second International Conference*. 2023, pp. 90–101.

[322] S. Cornell, Z.-Q. Wang, Y. Masuyama, S. Watanabe, M. Pariente, N. Ono, and S. Squartini. “Multi-Channel Speaker Extraction with Adversarial Training: The Wavlab Submission to The Clarity ICASSP 2023 Grand Challenge”. In: *Proc. of ICASSP*. IEEE. 2023, pp. 1–2.

[323] M. Ravanelli, T. Parcollet, P. Plantinga, A. Rouhe, S. Cornell, L. Lugosch, C. Subakan, N. Dawalatabad, A. Heba, J. Zhong, et al. “SpeechBrain: A general-purpose speech toolkit”. In: *arXiv preprint arXiv:2106.04624* (2021).

[324] S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishitoba, Y. Unno, N. E. Y. Soplin, J. Heymann, M. Wiesner, and N. e. a. Chen. “Espnet: End-to-end speech processing toolkit”. In: *arXiv preprint arXiv:1804.00015* (2018).

Appendices

Publications List

1 Journal Articles

1. S. Cornell, M. Omologo, S. Squartini, and E. Vincent. “Overlapped speech detection and speaker counting using distant microphone arrays”. In: *Computer Speech & Language* 72 (2022), p. 101306

2. C. Subakan, M. Ravanelli, S. Cornell, F. Grondin, and M. Bronzi. “On using transformers for speech-separation”. In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing (just accepted)* (2023)

2 Conference Articles

1. M. Severini, E. Principi, S. Cornell, L. Gabrielli, and S. Squartini. “Who Cried When: Infant Cry Diarization with Dilated Fully-Convolutional Neural Networks”. In: *International Joint Conference on Neural Networks (IJCNN).* 2020, pp. 1–8

2. S. Cornell, E. Principi, and S. Squartini. “A Novel Adversarial Training Scheme for Deep Neural Network based Speech Enhancement”. In: *International Joint Conference on Neural Networks (IJCNN).* 2020, pp. 1–8

3. M. Pariente, S. Cornell, A. Deleforge, and E. Vincent. “Filterbank design for end-to-end speech separation”. In: *Proc. of ICASSP.* 2020, pp. 6364–6368

4. S. Cornell, M. Omologo, S. Squartini, and E. Vincent. “Detecting and counting overlapping speakers in distant speech scenarios”. In: *Proc. of Interspeech.* 2020, pp. 3107–3111

5. M. Pariente, S. Cornell, J. Cosentino, S. Sivasankaran, E. Tzinis, J. Heitkaemper, M. Olvera, F.-R. Stöter, M. Hu, J. M. Martín-Doñas, et al. “Asteroid: the PyTorch-based audio source separation toolkit for researchers”. In: *Proc. of Interspeech.* 2020

6. S. Cornell, M. Olvera, M. Pariente, G. Pepe, E. Principi, L. Gabrielli, and S. Squartini. “Domain-adversarial training and trainable parallel front-end for the dcase 2020 task 4 sound event detection challenge”. In: *DCASE 2020-5th Workshop on Detection and Classification of Acoustic Scenes and Events*. 2020

7. S. Cornell, M. Olvera, M. Pariente, G. Pepe, E. Principi, L. Gabrielli, and S. Squartini. “Task-aware separation for the dcase 2020 task 4 sound event detection and separation challenge”. In: *DCASE 2020-5th Workshop on Detection and Classification of Acoustic Scenes and Events*. 2020

8. C. Subakan, M. Ravanelli, S. Cornell, M. Bronzi, and J. Zhong. “Attention is All You Need in Speech Separation”. In: *Proc. of ICASSP* (2020)

9. C. Aironi, S. Cornell, E. Principi, and S. Squartini. “Graph-based representation of audio signals for sound event classification”. In: *Proc. of EUSIPCO*. IEEE. 2021, pp. 566–570

10. G. Morrone, S. Cornell, E. Zovato, A. Brutti, and S. Squartini. “Conversational Speech Separation: an Evaluation Study for Streaming Applications”. In: *Proceedings of the 152nd International Audio Engineering Convention*. 2022

11. S. Cornell, A. Brutti, M. Matassoni, and S. Squartini. “Learning to rank microphones for distant speech recognition”. In: *Proc. of Interspeech*. 2021

12. F. Ronchini, R. Serizel, N. Turpault, and S. Cornell. “The impact of non-target events in synthetic soundscapes for sound event detection”. In: *DCASE 2021-Detection and Classification of Acoustic Scenes and Events*. 2021

13. C. Aironi, S. Cornell, E. Principi, and S. Squartini. “Graph Node Embeddings for ontology-aware Sound Event Classification: an evaluation study”. In: *Proc. of EUSIPCO*. IEEE. 2022, pp. 414–418

14. C. Subakan, M. Ravanelli, S. Cornell, and F. Grondin. “REAL-M: Towards speech separation on real mixtures”. In: *Proc. of ICASSP*. 2022, pp. 6862–6866

15. F. Ronchini, S. Cornell, R. Serizel, N. Turpault, E. Fonseca, and D. P. Ellis. “Description and analysis of novelties introduced in DCASE Task 4 2022 on the baseline system”. In: *DCASE 2022-7th Workshop on Detection and Classification of Acoustic Scenes and Events* (2022)

16. Y.-J. Lu, S. Cornell, X. Chang, W. Zhang, C. Li, Z. Ni, Z.-Q. Wang, and S. Watanabe. "Towards low-distortion multi-channel speech enhancement: The ESPNet-SE submission to the L3DAS22 challenge". In: *Proc. of ICASSP*. 2022, pp. 9201–9205

17. Y.-J. Lu, X. Chang, C. Li, W. Zhang, S. Cornell, Z. Ni, Y. Masuyama, B. Yan, R. Scheibler, Z.-Q. Wang, et al. "ESPnet-SE++: Speech enhancement for robust speech recognition, translation, and understanding". In: *Proc. of ICASSP* (2022)

18. S. Cornell, M. Pariente, F. Grondin, and S. Squartini. "Learning filterbanks for end-to-end acoustic beamforming". In: *Proc. of ICASSP*. IEEE. 2022, pp. 6507–6511

19. Z.-Q. Wang, S. Cornell, S. Choi, Y. Lee, B.-Y. Kim, and S. Watanabe. "TF-GridNet: Integrating Full-and Sub-Band Modeling for Speech Separation". In: *Proc. of ICASSP* (2022)

20. Z.-Q. Wang, S. Cornell, S. Choi, Y. Lee, B.-Y. Kim, and S. Watanabe. "Neural Speech Enhancement with Very Low Algorithmic Latency and Complexity via Integrated full-and sub-band Modeling". In: *Proc. of ICASSP*. IEEE. 2023, pp. 1–5

21. R. Serizel, S. Cornell, and N. Turpault. "Performance Above All? Energy Consumption vs. Performance, a Study on Sound Event Detection with Heterogeneous Data". In: *Proc. of ICASSP*. IEEE. 2023, pp. 1–5

22. C. Aironi, S. Cornell, and S. Squartini. "Tackling the Linear Sum Assignment Problem with Graph Neural Networks". In: *Applied Intelligence and Informatics: Second International Conference*. 2023, pp. 90–101

23. Y. Masuyama, X. Chang, S. Cornell, S. Watanabe, and N. Ono. "End-to-End Integration of Speech Recognition, Dereverberation, Beamforming, and Self-Supervised Learning Representation". In: *Spoken Language Technology Workshop*. 2023, pp. 260–265

24. G. Morrone, S. Cornell, D. Raj, L. Serafini, E. Zovato, A. Brutti, and S. Squartini. "Low-Latency Speech Separation Guided Diarization for Telephone Conversations". In: *Spoken Language Technology Workshop*. 2023, pp. 641–646

25. S. Cornell, T. Balestri, and T. Sénéchal. "Implicit acoustic echo cancellation for keyword spotting and device-directed speech detection". In: *Spoken Language Technology Workshop*. 2023, pp. 1052–1058

26. S. Cornell, Z.-Q. Wang, Y. Masuyama, S. Watanabe, M. Pariente, N. Ono, and S. Squartini. “Multi-Channel Speaker Extraction with Adversarial Training: The Wavlab Submission to The Clarity ICASSP 2023 Grand Challenge”. In: *Proc. of ICASSP*. IEEE. 2023, pp. 1–2

Other Contributions

1 Open Source Software Contributions

1. Asteroid: The PyTorch-based audio source separation toolkit for researchers [262], available at github.com/asteroid-team/asteroid.
2. SpeechBrain: A general-purpose speech toolkit [323], available at github.com/speechbrain/speechbrain.
3. DCASE Task 4 baseline [318], available at github.com/DCASE-REPO/DESED_task/.
4. ESPNet2 [25, 324] available at github.com/espnet/espnet.
5. CHiME-7 DASR baseline, available at github.com/espnet/espnet/tree/master/egs2/chime7_task1.

2 Organization

1. Team leader of the DCASE 2020 Task 4 joint UNIVPM/Inria team.
2. Co-organizer of DCASE 2021 and 2022 Task 4 Challenge [318].
3. Co-organizer of ICASSP 2023 special session on “resource-efficient real-time neural speech enhancement”.
4. Main organizer for the CHiME-7 DASR Challenge, available at www.chimechallenge.org/current/task1/index.