**BRNO UNIVERSITY OF TECHNOLOGY**
**VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ**

**FACULTY OF INFORMATION TECHNOLOGY**
**FAKULTA INFORMAČNÍCH TECHNOLOGIÍ**

**DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA**
**ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ**


# FAR-FIELD SPEAKER VERIFICATION INCORPORATING MULTI-CHANNEL PROCESSING

**ROZPOZNÁVÁNÍ MLUVČÍHO ZE VZDÁLENÉHO ZDROJE S VÍCEKANÁLOVÝM ZPRACOVÁNÍM**

**AUDIA**

**PHD THESIS**
**DISERTAČNÍ PRÁCE**


**AUTHOR** — **Ing. LADISLAV MOŠNER**
**AUTOR PRÁCE**

**SUPERVISOR** — **prof. Dr. Ing. JAN ČERNOCKÝ**
**ŠKOLITEL**

**CO-SUPERVISOR** — **Ing. OLDŘICH PLCHOT, Ph.D.**
**ŠKOLITEL SPECIALISTA**



**BRNO 2025**

# Abstract

Far-field speech processing has gained increasing attention in recent years with the advent of smart speakers, home assistants, and meeting transcription systems. To support these applications, robust far-field speech processing techniques are required. A key task enabling personalized interaction is speaker verification. Compared to close-talking conditions, far-field systems face additional challenges such as reverberation and background noise, which degrade the target speech. To mitigate these effects, far-field devices typically employ microphone arrays that provide spatial information. These challenges and opportunities motivate this thesis, focusing on multi-channel speaker verification.

Despite significant progress in related fields of speech processing, multi-channel speaker verification remains underexplored, hindered by limited data resources and specialized techniques. This thesis focuses on both aspects. On the data side, we repurposed existing publicly available corpora and created the MultiSV dataset, which provides simulated multi-channel mixtures with speech/noise training targets and speaker labels. MultiSV also defines multiple evaluation protocols based on retransmitted recordings, supporting various scenarios, such as single clean versus multi-channel corrupted enrollment. To support training more data-demanding models, we further introduced an extended dataset, MultiSV2.

On the modeling side, we first approached multi-channel speaker embedding extraction using a cascaded strategy, decomposing the problem into multi-channel preprocessing and single-channel embedding extraction. Motivated by advances in speech separation, we designed models ranging from signal-processing-based methods to hybrid neural network and beamforming front-ends. Notably, we proposed direct and indirect mask prediction for mask-based beamforming, and the reference-channel attention (RCA) combiner, which generalizes single-channel separation models to multi-channel inputs.

Recognizing the limitations of cascaded models, such as error propagation and different objectives of the modules, we next explored unified architectures for multi-channel embedding extraction. Leveraging MultiSV2, we fine-tuned cascaded components jointly with the end-task loss, and subsequently introduced METRO, a general framework that extends self-supervised speech representation models to multi-channel settings. METRO yields multi-channel speaker embeddings. However, it is general and potentially applicable to other speech processing tasks.



# Abstrakt

Zpracování řeči ze vzdálených zdrojů upoutává v posledních letech stále větší pozornost díky vzestupu inteligentních reproduktorů, domácích asistentů nebo systémů pro přepisování schůzek. K podpoře těchto aplikací jsou zapotřebí robustní techniky pro zpracování řeči ze vzdálených zdrojů. Klíčem k personalizované interakci je verifikace mluvčího. Ve srovnání se zpracováním signálů získaných mikrofony umístěnými blízko zdrojů čelí systémy zpracovávající nahrávky ze vzdálených senzorů dalším výzvám. Jimi jsou jevy, které zhoršují kvalitu cílové řeči, zejména dozvuk a šum v pozadí. Zařízení pro vzdálené snímání proto často využívají mikrofonní pole, která mohou napomoci zmírnění uvedených jevů díky tomu, že poskytují informaci o prostoru. Nastíněné výzvy a příležitosti motivují tuto práci, jež se zaměřuje na vícekanálovou verifikaci mluvčího.

Z důvodu omezených zdrojů dat zůstává vícekanálová verifikace mluvčích nedostatečně prozkoumaná, a to navzdory významnému pokroku v souvisejících oblastech zpracování

řeči. Předložená práce se věnuje dvěma fundamentálním aspektům: absenci dat i specializovaným technikám zpracování. V rámci problematiky dat jsme přetvořili existující veřejně dostupné datové sady a vyvinuli novou sadu, MultiSV, která poskytuje simulované vícekanálové směsi signálů s trénovacími referenčními signály řeči/šumu a identitami mluvčích. MultiSV rovněž obsahuje znovu přenesené nahrávky využitelné pro vyhodnocování verifikačních systémů s podporou různých scénářů. Příkladem může být varianta s registrací do systému pomocí jednokanálových čistých nebo vícekanálových poškozených nahrávek. Abychom podpořili trénování modelů s vyššími nároky na data, vytvořili jsme rozšířenou datovou sadu MultiSV2.

Na úrovni modelů jsme nejprve přistoupili k extrakci řečových embeddingů z vícekanálového audia pomocí kaskádové strategie, přičemž jsme problém rozložili na vícekanálové předzpracování a jednokanálovou extrakci embeddingů. Motivováni pokroky v oblasti separace řeči jsme navrhli různé techniky od těch založených na zpracování signálů po hybridní, které kombinují neuronové sítě a beamforming. V práci se diskutuje přímá a nepřímá predikce masek pro beamforming na nich založený. Dále je představena metoda RCA (reference channel attention), která zobecňuje jednokanálové separační modely tak, aby mohly využít vícekanálové vstupy.

S uvědoměním si omezení kaskádových modelů, jimiž jsou šíření chyb a odlišné objektivní funkce jednotlivých modulů, jsme dále prozkoumali jednotné architektury pro vícekanálovou extrakci embeddingů. S využitím MultiSV2 jsme dokázali ladit parametry spojených komponent kaskádových modelů s využitím cílové objektivní funkce. Následně jsme navrhli techniku METRO (multi-channel extension of pre-trained models), která rozšiřuje původně jednokanálové modely pro reprezentaci řeči trénované pomocí samořídícího učení na vícekanálové. Ačkoli METRO v kombinaci s vhodným extrakčním modelem produkuje řečové embeddingy pro vícekanálové audio, metoda samotná je obecná a potenciálně využitelná v dalších oblastech zpracování řeči.

## Keywords

multi-channel speaker verification, microphone arrays, beamforming, speech separation, speaker embedding extraction, MultiSV

## Klíčová slova

vícekanálová verifikace mluvčích, mikrofonní pole, beamforming, separace řeči, extrakce embeddingů mluvčích, MultiSV

## Reference

MOŠNER, Ladislav. *Far-Field Speaker Verification Incorporating Multi-Channel Processing.* Brno, 2025. PhD thesis. Brno University of Technology, Faculty of Information Technology. Supervisor prof. Dr. Ing. Jan Černocký. Co-supervisor Ing. Oldřich Plchot, Ph.D.

# Rozšířený abstrakt

Zpracování řeči ze vzdálených zdrojů upoutává v posledních letech stále větší pozornost díky vzestupu inteligentních reproduktorů, domácích asistentů a systémů pro přepisování schůzek. K podpoře těchto aplikací jsou zapotřebí robustní techniky pro zpracování řeči ze vzdálených zdrojů. Klíčem k personalizované interakci je verifikace mluvčího. Ve srovnání se zpracováním signálů získaných mikrofony umístěnými blízko zdrojů čelí systémy zpracovávající nahrávky ze vzdálených senzorů dalším výzvám. Jimi jsou dva jevy, které zhoršují kvalitu cílové řeči, dozvuk a šum v pozadí. Zařízení pro vzdálené snímání proto často používají mikrofonní pole, která mohou napomoci zmírnění uvedených jevů díky tomu, že poskytují informaci o prostoru. Nastíněné výzvy a příležitosti motivují tuto práci, jež se zaměřuje na vícekanálovou verifikaci mluvčího.

Z důvodu omezených zdrojů dat zůstává vícekanálová verifikace mluvčích nedostatečně prozkoumaná, a to navzdory významnému pokroku v souvisejících oblastech zpracování řeči. Předložená práce se věnuje dvěma fundamentálním aspektům: absenci dat i specializovaným technikám zpracování. V rámci problematiky dat jsme přetvořili existující veřejně dostupné datové sady a vytvořili novou sadu, MultiSV, která poskytuje simulované vícekanálové směsi signálů s trénovacími referenčními signály řeči/šumu a identitami mluvčích. MultiSV rovněž obsahuje znovu přenesené nahrávky využitelné pro vyhodnocování verifikačních systémů s podporou různých scénářů. Příkladem může být varianta s registrací do systému pomocí jednokanálových čistých nebo vícekanálových poškozených nahrávek. Abychom podpořili trénování modelů s vyššími nároky na data, vytvořili jsme rozšířenou datovou sadu MultiSV2.

Sběr reálných vícekanálových dat obsahujících identity mluvčích v množství, které je třeba pro trénování extraktorů embeddingů, je velmi náročné na finanční i lidské zdroje. Z toho důvodu jsme v první fázi našich experimentů rozdělili daný problém na dva podproblémy, které je snazší řešit právě s ohledem na dostupná trénovací data. 1) První podproblém sestává z vícekanálového zpracování, kde výsledkem je jednokanálová nahrávka s potlačeným šumem a ponechanou řečí. 2) Druhým podproblémem je extrakce řečových embeddingů, přičemž vstupem je signál zbavený šumu. Pro trénování takových jednokanálových extraktorů jsou data dostupná. Spojením komponent pro první a druhý podproblém dostáváme kaskádový systém. První problém, se kterým se musí systémy zpracovávající nahrávky pořízené ze vzdálených mikrofonů potýkat, je dozvuk. V raných fázích našeho výzkumu jsme se tak zabývali výlučně dopady dozvuku na přesnost verifikace a způsoby jeho potlačení. Podle našich experimentů na datech, která nám byla k dispozici (BUT ReverbDB), jsme zjistili, že efekt dozvuků může být dramatický a zhoršit výsledky více než 6krát. Za využití vícekanálového signálového zpracování WPE (weighted prediction error) pro potlačení dozvuku, beamformingu v rámci předzpracování a pečlivého výběru dat pro trénování komponenty poskytující verifikační skóre jsme dokázali ztrátu přesnosti systémů téměř vymazat. To nás motivovalo k řešení těžšího a reálnějšího problému, kdy je kromě dozvuku verifikace ztížená přítomností rušivého zdroje (jenž obecně nazýváme šumem). Navrhli jsme několik kaskádových modelů, které lze jednotně popsat: V rámci modulu pro předzpracování jsme využili neuronovou síť predikující řeč a šum na jisté úrovni abstrakce. Predikce jsou využity pro odhad kombinačních vektorů pro beamforming (MVDR nebo GEV). Modely použité pro separaci zdrojů byly původně jednokanálové, tudíž musely být aplikované na každý kanál nezávisle. Pro využití signálů z ostatních mikrofonů již na úrovni separace jsme navrhli metodu RCA (reference channel attention), která rozšiřuje mnohé modely produkující signály v časové doméně na vícekanálové. Na úrovni extrakce embeddingů jsme využily různých modelů opět založených na neuronových sítích. V práci

se zabýváme analýzou různých instancí nastíněného komplexního systému a vlivem komponent (včetně RCA) na výsledky. Ačkoli jsme dosáhli významného zlepšení v porovnání s výsledky, které lze získat pouze s jedním kanálem, kaskádový systém trpí několika problémy. Prvním z nich je skutečnost, že chyby na úrovni konkrétního modulu se propagují dál do systému a kumulují. Druhým problémem je, že jak modul předzpracování, tak i extraktor embeddingů optimalizují různá kritéria. Pro zmírnění těchto nedostatků jsme využili rozšířenou datovou sadu, MultiSV2, která obsahuje informaci o identitách mluvčích. Mohli jsme tak spojit oba moduly a ladit parametry jednoho nebo obou s využitím cílové objektivní funkce. Experimentovali jsme s několika přístupy, kde jako nejlepší se ukázal ten, kdy se nejprve ladí parametry pouze extraktoru a posléze celého spojeného modelu. Tento přístup má potenciál zlepšit výsledky až o 30 % relativně v závislosti na datech.

Přes dosažené zlepšení je výsledný model stále kaskádový. Naším cílem bylo vytvořit vícekanálový model pro extrakci embeddingů bez explicitního rozdělení na předzpracování a jednokanálovou extrakci. K němu nám dopomohla právě datová sada MultiSV2 a současně dostupnost modelů pro reprezentaci řeči předtrénovaných pomocí samořídicího učení. Navrhli jsme techniku METRO (multi-channel extension of pre-trained models), která rozšiřuje zmíněné předtrénované modely na vícekanálové. Ty jsme následně použili společně s připojeným malým trénovatelným modulem pro extrakci embeddingů. Výsledný model překonal náš nejlepší kaskádový model o více než 25 % relativně. Technika METRO navíc není navržená výlučně pro vícekanálovou verifikaci mluvčích. V našem plánu do budoucna je tak zkoumání jejího využití v dalších oblastech zpracování řeči.

# Far-Field Speaker Verification Incorporating Multi-Channel Processing

## Declaration

I hereby declare that this doctoral dissertation was prepared as an original work by the author under the supervision of prof. Jan Černocký and Dr. Oldřich Plchot. I have listed all the literary sources, publications and other sources, which were used during the preparation of this thesis.

.....
Ladislav Mošner
August 31, 2025


## Acknowledgements

Throughout my studies, I was grateful and honored to call myself a member of Brno Speech@FIT, a world-class speech processing research group. It would not have existed without the head and supervisor of my thesis, Honza Černocký. I owe my deepest gratitude to him for being an excellent leader and kind supervisor. I am very grateful for all the support of my investigations. He allowed me to undertake multiple internships and always gave me invaluable feedback on my manuscripts. His deep expertise and enthusiasm encouraged me to pursue doctoral studies.

This work would not have been possible without the tremendous support of my co-supervisor, Olda Plchot, who introduced me to the world of speaker verification and kindly shared his knowledge with me. I am profoundly grateful for his highly valued guidance, constructive feedback, countless meetings, and writing proposals to support our research. During our evening coffee sessions, I always enjoyed learning about gardening or other topics.

My sincere thanks go to my colleagues at Brno Speech@FIT who drive the group with their deep knowledge, expertise, cutting-edge ideas, and curiosity, creating a motivating yet welcoming environment. I wish to express my sincere appreciation to Lukáš Burget, who immensely enriches the speech group with his extraordinary knowledge, enthusiasm, dedication, helpfulness, and warm attitude. And coffee mastery. I would like to thank all past and current members of our speaker identification subgroup, Anya Silnova, Johan Rohdin, Junyi Peng, Fede Landini, and Ondra Glembek, for our discussions in meetings, and Pavel Matějka for his past leadership. My thanks also go to past and current officemates, Katka Žmolíková, Karel Beneš, Bolaji Yusuf, Alex Polok, and Šimon Sedláček, and my past classmate and roommate, Martin Šůstek.

I am deeply grateful to all my colleagues, collaborators, and co-authors outside Brno Speech@FIT. My special thanks go to Themos Stafylakis for his expertise, inspiration, and ideas, as well as for our great collaboration. I would also like to thank all the collaborators on NIST evaluations, especially Sara Barahona, for their dedication and great work.

I am profoundly grateful to my family, especially my mum, dad, sister, and brother, for their immense support, encouragement, and understanding throughout all the years.

I am deeply indebted to my wife, whom I dedicate this thesis. I am profoundly grateful for her love, patience, and sustained support.

# Contents


**1 Introduction** **8**
- 1.1 Contributions . . . 9
- 1.2 Organization of the Thesis . . . 11

**2 Speaker Verification** **12**
- 2.1 Task Definition . . . 12
- 2.2 Evaluation Metrics . . . 13
  - 2.2.1 Detection Error Tradeoff Curve . . . 14
  - 2.2.2 Equal Error Rate . . . 14
  - 2.2.3 Detection Cost Function . . . 14
- 2.3 Speaker Verification Evolution . . . 15
  - 2.3.1 The i-vector Approach . . . 15
  - 2.3.2 The Era of Neural Networks . . . 18
- 2.4 Modern Speaker Verification . . . 19
  - 2.4.1 Speaker Embedding Extraction . . . 19
  - 2.4.2 Probabilistic Linear Discriminant Analysis . . . 26
  - 2.4.3 Cosine Similarity Scoring . . . 30

**3 Microphone Array Signal Processing and Spatial Filtering** **31**
- 3.1 Beamforming Theory . . . 32
  - 3.1.1 Continuous Apertures . . . 33
  - 3.1.2 Microphone Arrays . . . 36
  - 3.1.3 Beamforming . . . 38
- 3.2 Adaptive Beamforming . . . 41
  - 3.2.1 Signal Model . . . 41
  - 3.2.2 Minimum Variance Distortionless Response . . . 42
  - 3.2.3 Generalized Eigenvalue Beamformer . . . 46
- 3.3 Broadband Beamforming . . . 48
  - 3.3.1 Signal Model . . . 49

**4 Corpora** **51**
- 4.1 VoxCeleb . . . 51
- 4.2 Retransmitted BUT RevertbDB Corpus . . . 53
- 4.3 MultiSV Dataset . . . 55
  - 4.3.1 Data Situation Before MultiSV . . . 55
  - 4.3.2 Analysis of the VOiCES Corpus . . . 56
  - 4.3.3 MultiSV Development and Evaluation Data . . . 59
  - 4.3.4 MultiSV Training Data . . . 61

4.4 Extension of MultiSV — MultiSV2 . . . 62
4.4.1 Noise Resources . . . 63

**5 Speech Separation for Speaker Verification 65**
5.1 Single-Channel Speech Separation Overview . . . 65
5.1.1 Speech Enhancement . . . 65
5.1.2 Speaker Separation . . . 68
5.2 Multi-Channel Speech Separation Overview . . . 69
5.3 Evaluation Metrics . . . 70
5.3.1 Subjective Evaluation . . . 71
5.3.2 Objective Evaluation . . . 71
5.4 Oracle Beamforming and Hyperparameters . . . 74
5.5 Separation Models . . . 76
5.5.1 Recurrent Mask Estimator . . . 76
5.5.2 Time-Domain Separation Models . . . 76
5.6 Using Outputs of Separation Models for Beamforming . . . 80
5.6.1 Direct Mask Prediction for Mask-Based Beamforming . . . 80
5.6.2 Indirect Mask Prediction for Mask-Based Beamforming . . . 83
5.6.3 Mask-Less Beamforming . . . 85
5.7 Multi-Channel Speech Separation with Cross-Attention and Beamforming . 86
5.7.1 Reference Channel Attention Combiner . . . 87
5.7.2 Extension of Well-Known Single-Channel Separation Models . . . 90
5.7.3 Experiments With Reference Channel Attention . . . 90
5.8 Summary . . . 94

**6 Multi-channel Speaker Verification 95**
6.1 Task Definition . . . 95
6.2 Approaches to Multi-Channel Speaker Verification . . . 96
6.3 Multi-Channel Dereverberation and I-vectors . . . 98
6.3.1 Pre-Processing and Adaptation . . . 98
6.3.2 Speaker Verification System Description . . . 99
6.3.3 Experimental Setup . . . 100
6.3.4 Evaluation of Dereverberation and Backend Adaptation . . . 102
6.3.5 Conclusions of the Experiments . . . 105
6.4 Multi-Channel Speaker Verification via Cascaded Beamforming and Speaker Embedding Extraction . . . 106
6.4.1 General Framework . . . 106
6.4.2 Outline of the Framework Development . . . 108
6.4.3 Multi-Channel Speaker Verification with Application-Aware Beamforming and PLDA . . . 108
6.4.4 Multi-Channel Speaker Verification With Conv-TasNet-Based Beamformer . . . 115
6.4.5 Framework Analysis: Front-Ends and Joint Fine-Tuning . . . 119
6.5 Summary . . . 129

**7 Multi-Channel Speaker Verification Building on Foundation Speech Models 131**
7.1 Foundation Models and Multi-Channel Data . . . 131

7.2 Multi-Channel Extension of Pre-Trained Models . . . 132
7.2.1 Method . . . 132
7.2.2 M-M Modules . . . 133
7.2.3 M-S Modules . . . 135
7.3 Experiments . . . 135
7.3.1 Experimental Setup . . . 135
7.3.2 Towards the Multi-Channel Extension of the SSL Models . . . 136
7.3.3 Layer Dependence . . . 138
7.4 Summary . . . 138

**8 Conclusions** **140**
8.1 Future Work . . . 142

**Bibliography** **144**

**A I-vector Derivations** **165**
A.1 Data Likelihood Under the GMM Model . . . 165
A.2 Data Alignment Using Universal Background Model . . . 167
A.3 I-Vector Posterior . . . 168

**B Derivations for Simplified PLDA** **170**
B.1 Generative Equations . . . 170
B.2 Covariance Matrix Inversion . . . 170
B.3 Score Computation with the Simplified PLDA . . . 172

# Glossary

**AAM** additive angular margin (softmax). 26, 30, 120, 127, 136

**AM** additive margin (softmax). 26, 30, 117, 118

**ASR** automatic speech recognition. 8, 53, 56, 74, 75, 80, 83, 96, 97, 107, 109, 139, 143

**BAN** blind analytical normalization. 48

**BCE** binary cross-entropy. 76, 109, 113–115

**CASA** computational auditory scene analysis. 65

**CE** cross-entropy. 114, 115

**CMVN** cepstral mean and variance normalization. 99

**D50** definition acoustic parameter. 99, 102, 103

**DCF** detection cost function. 14, 15

**DET** detection error tradeoff. 14

**DFT** discrete Fourier transform. 48, 73

**DNN** deep neural network. 19, 67

**DR30** Doddington's Rule of 30. 61

**DTFT** discrete-time Fourier transform. 41, 42

**DoA** direction of arrival. 32, 33

**E-TDNN** extended time-delay neural network. 21, 129

**ECAPA-TDNN** emphasized channel attention, propagation and aggregation in time-delay neural network. 21, 24, 97

**EER** equal error rate. 14, 18, 52, 102, 103, 105, 113, 114, 119, 121, 122, 126, 127, 136

**EM** expectation–maximization. 18, 27

**FAR** false acceptance (alarm) rate. 14

**FIR** finite impulse response. 72

**GCC-PHAT** generalized cross-correlation with phase transform. 99, 104

**GEV** generalized eigenvalue (beamformer). 10, 11, 41, 46–48, 50, 69, 97, 99, 100, 108–110, 113, 116, 122, 129, 142

**GMM** Gaussian mixture model. 16, 17, 99

**GMVDR** generalized minimum variance distortionless response (beamformer). 47

**HT-PLDA** heavy-tailed probabilistic linear discriminant analysis. 27

**IBM** ideal binary mask. 65, 67, 70, 75, 76, 82, 83, 109, 113, 115, 117, 121, 122

**IRM** ideal ratio mask. 67, 70, 75, 76, 82–84, 109, 121, 122, 124

**ISM** image-source method. 60, 61, 63

**JFA** joint factor analysis. 15, 16

**LDA** linear discriminant analysis. 100, 112

**LLR** log-likelihood ratio. 28

**LSTM** long short-term memory. 74, 76, 77, 122, 125

**MAP** maximum a posteriori. 15

**METRO** multi-channel extension of pre-trained models. 11, 138, 139, 143

**MFCC** Mel-frequency cepstral coefficients. 21, 99, 110

**MHFA** multi-head factorized attentive (pooling). 24, 25, 132, 133, 136, 137, 139

**MOS** mean opinion score. 71, 74

**MR** miss rate. 14

**MSE** mean squared error. 67, 68, 76, 100, 109, 114, 115

**MVDR** minimum variance distortionless response (beamformer). 11, 41–48, 50, 65, 69, 74, 80, 82, 92, 97, 108, 116, 119, 129, 137, 142

**MWF** multi-channel Wiener filter. 69, 97

**NIST** National Institute of Standards and Technology. 14, 22, 53–55, 101, 140

**PESQ** perceptual evaluation of speech quality. 72, 74

**PIT** permutation invariant training. 69, 79, 116

**PLDA** probabilistic linear discriminant analysis. 15, 18, 19, 26–29, 95, 98–106, 108, 112, 116, 129

**POLQA** perceptual objective listening quality assessment. 72

**PReLU** parametric rectified linear unit. 78

**PSDA** probabilistic spherical discriminant analysis. 19

**PSD** power spectral density. 43

**PSM** phase-sensitive mask. 67

**PSVM** pairwise support vector machine. 19

**RCA** reference channel attention. 10, 86–90, 92–94, 125, 126, 128–130, 141

**RIR** room impulse response. 53, 60–63, 97, 100, 101, 103, 113

**RNN** recurrent neural network. 68, 79

**RTF** relative transfer function. 42, 49

**RT** reverberation time. 53, 55, 56, 60, 62, 64, 113

**ReLU** rectified linear unit. 21, 76, 80, 87

**SCM** spatial covariance matrix. 43–46, 69, 70, 75, 76, 80–82, 84, 85, 93, 97, 110, 115–117, 119, 122, 124, 128, 141

**SDRi** signal-to-distortion ratio improvement. 73, 75, 81–86, 93, 122, 124, 127, 130, 142

**SDR** signal-to-distortion ratio. 72

**SE** squeeze-and-excitation. 120

**SI-SDR** scale-invariant signal-to-distortion ratio. 72, 74, 116

**SI-SNR** scale-invariant signal-to-noise ratio. 72, 79

**SNR** signal-to-noise ratio. 46, 47, 59–62, 64, 67, 100, 113, 127

**SRE** speaker recognition evaluation. 14, 54, 55

**SSL** self-supervised learning. 9, 11, 23, 24, 107, 131–133, 136–138, 142, 143

**STFT** short-time Fourier transform. 48, 49, 69, 70, 74, 84, 85, 92, 93, 100, 107, 110, 114, 116, 120, 141

**STOI** short-time objective intelligibility. 72–75, 81–86, 93, 122, 124

**SUPERB** speech processing universal performance benchmark. 23, 24, 135

**SV** speaker verification. 8–14, 16, 18, 19, 23, 26, 27, 51–58, 61, 62, 65, 70, 80, 86, 94–99, 101–103, 105, 107, 108, 115–117, 119, 121, 124–126, 130–133, 135, 136, 138–143

**T-PSDA** toroidal probabilistic spherical discriminant analysis. 19

**TAP** temporal average pooling. 20, 24

**TCN** temporal convolutional network. 77, 78

**TDNN** time-delay neural network. 20, 21, 23, 24, 108, 116, 117

**TDOA** time difference of arrival. 99, 100

**UBM** universal background model. 16, 17, 99

**UCA** uniform circular array. 31

**ULA** uniform linear array. 31, 36, 61

**VAD** voice activity detection. 19, 22, 44, 45, 62

**VLAD** vector of locally aggregated descriptors. 20

**WPE** weighted prediction error. 98, 100, 103–105, 129

# Chapter 1

# Introduction

With the advent of hands-free devices, such as smart speakers, home assistants, and automatic meeting transcription systems, far-field speech processing has become an urgent research topic. Tasks such as automatic speech recognition (ASR), speech enhancement, and source separation have attracted significant attention. Although initially overlooked, the extraction of speaker representations (*embeddings*) eventually stood out for its unique role. It allows for assigning identities to utterances through diarization techniques and personalized content/responses. Speaker verification (SV) systems lie at the heart of such applications, and they form the core of this thesis. Despite benefits for users, distant speech capturing still poses challenges to systems processing the recordings. The data acquisition process usually takes place in rooms, which stems from use cases. As a result, it is not only attenuated speech that is recorded. Owing to sound reflections from walls and other obstacles, reverberation is one of the causes that hinders the performance of far-field speech processing models. Another important detrimental aspect is the presence of distractors. A variety of interference sources may be broad, ranging from diffuse to localized noises. The sources of noise can be diverse, manifesting different characteristics. Television, loudspeakers, vents, and undesired speech sources are just some of these distractors.

The adverse room conditions call for the robustness of distant speech processing systems. Microphone arrays are usually adopted with the intention of improving robustness. Therefore, far-field-operating devices are commonly equipped with numerous sensors. They provide spatial information, helping to combat the harmful effects of noise sources and reverberation. Research on the utilization of multiple audio channels in various tasks has been actively conducted in line with this trend. Our specific target was using microphone arrays for SV, which is termed *multi-channel speaker verification*.

Although the task at hand is appealing for the potential applications outlined before, it is still a niche field of speech processing, as studies on this topic are relatively scarce compared to other tasks employing multiple channels. We aimed to contribute to the community by fostering research in this field. To this end, we worked on two frontiers. 1) Since multi-channel speaker verification is an underexplored area, there are limited corpora for development. Compared to single-channel data, which is commonly scraped from the internet, obtaining realistic multi-channel data with speaker labels is much more difficult and, in turn, expensive. The scarcity of data then limits the interest in the community. With the hope of alleviating such problems, we worked toward developing a dataset enabling both training and evaluation, setting a common testbed that could ease comparison of systems and competition. Our efforts resulted in a publicly available dataset, MultiSV, and its second version, extending training data, MultiSV2. 2) We attempted to advance the field by

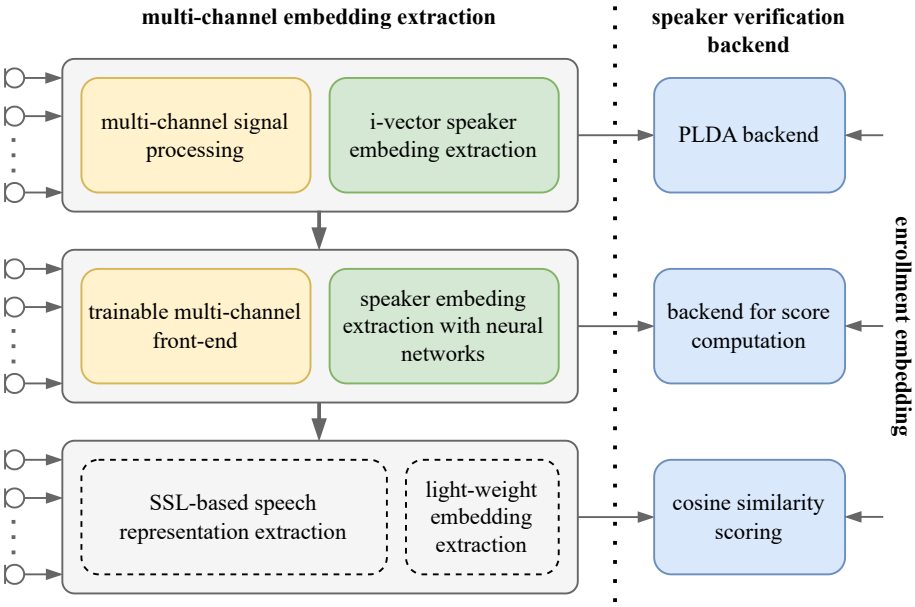



Figure 1.1: The evolution of the proposed systems targeting a holistic approach to speaker embedding extraction using multiple microphones.

designing and investigating models for multi-channel SV. Inspired by the benefits of multi-channel processing, we initially built various cascaded systems as shown in the top part of Figure 1.1. The common issue of cascaded systems (not only in multi-channel SV but also in diarization and other fields) is that the components optimize different criteria in the first place. Moreover, errors from individual sub-systems are propagated to the following ones, having a cumulative effect. Therefore, we aimed to develop an end-to-end speaker embedding extraction. The first step towards this goal was developing a modular system with differentiable components that can be pre-trained separately and then fine-tuned with the common speaker-discriminative objective (see the middle part of Figure 1.1). Finally, we achieved an end-to-end architecture (including multi-channel processing without explicit multi-channel pre-processing) by employing speech representation models pre-trained via self-supervised learning (SSL). It is schematically presented in the bottom part of Figure 1.1.

## 1.1 Contributions

The core of the thesis is based on our related publications [Mošner et al., 2018a, Mošner et al., 2018b, Mošner et al., 2019, Mošner et al., 2022a, Mošner et al., 2023, Mošner et al., 2020, Mošner et al., 2022b, Mošner et al., 2024, Szőke et al., 2019]. On top of these studies, we provide more analyses, especially on front-ends in the cascaded model and joint fine-tuning with the embedding extraction. Our overall contributions can be summarized as follows:

- The beginning of our multi-channel data-related efforts is related to the BUT ReverbDB corpus [Szőke et al., 2019] available at `https://speech.fit.vut.cz/software/but-speech-fit-reverb-database`.

It provided the community with retransmitted multi-channel evaluation data for both automatic speech recognition and speaker verification. In return, it attracted considerable attention, reflected by the number of citations. We employed it in this thesis, as well as in [Mošner et al., 2018b].

- Being concerned about the data situation in the field of multi-channel speaker verification, a lack of common training and evaluation data, we tackled the issue by designing a novel dataset. Inspired by the properties of the VOiCES corpus [Richey et al., 2018], we analyzed its utility in multi-channel SV by redefining verifications trials in [Mošner et al., 2020]. This work was a foundation for our corpus. We then extended the previous study and defined multiple protocols for various use cases. Extending the evaluation data (comprising four protocols) with simulated training data, we published the MultiSV dataset in [Mošner et al., 2022b]. Our goal was to foster research in the field; therefore, we prepared descriptions and scripts to recreate the dataset from publicly available data at `https://github.com/BUTSpeechFIT/MultiSV`.

- The training dataset defined by MultiSV is useful particularly for training the front-ends of cascaded models, as it provides a limited number of examples. In order to allow training of data-demanding multi-channel embedding extractors and fine-tuning of cascaded systems, we created an extended version of MultiSV training data called MultiSV2 [Mošner et al., 2025]. We also provide simulation parameters and details to create the dataset at `https://github.com/BUTSpeechFIT/MultiSV/tree/main/MultiSV2`.

- The first problem that arises when dealing with far-field data and hinders the performance of speech processing models is reverberation. We devoted studies [Mošner et al., 2018a, Mošner et al., 2018b] to this issue. As a result, we devised a system comprising multi-channel processing and employing carefully selected data for back-end training. The final results approached those obtained on clean data with the same underlying model architecture. Hence, we showed that the problems with reverberation in multi-channel SV can be effectively alleviated.

- At the time of our experiments with mask-based beamformers, serving as a front-end in cascaded systems, the differentiability of the generalized eigenvalue problem required by models using the GEV beamformer was not supported by standard machine learning libraries. We formulated a new approach by altering the generalized eigenvalue problem via Cholesky decomposition, whose differentiability was widely supported. This modification allowed us to fine-tune front-end models in cascaded systems by back-propagating error from the embedding extractor. We termed this approach application-aware beamforming [Mošner et al., 2019].

- Extending the cascaded systems with (at that time) state-of-the-art time-domain speech separators, we designed a model where the front-end employs mask-based frequency domain beamforming and time-domain separation inspired by [Ochiai et al., 2020]. To this end, we employed an indirect mask prediction in [Mošner et al., 2022a].

- In the context of speech separation for front-ends in cascaded systems, we extended various time-domain models to their multi-channel versions. Our proposed integration of a new module, reference channel attention (RCA) combiner, into existing networks, making them capable of processing multi-channel inputs, was presented in [Mošner et al., 2023].

- Departing from cascaded systems, we proposed an approach to extend speech representation models trained in a self-supervised learning (SSL) fashion to the microphone arrays setting [Mošner et al., 2024]. We termed this approach METRO (Multi-channel ExTension of pRe-trained mOdels) and provided a relevant code at https://github.com/BUTSpeechFIT/Wespeaker_MC_SSL. We note that we were concerned with multi-channel SV. However, the approach is not necessarily specific to speaker verification.

Other publications loosely related or unrelated to the main topic of this thesis are available on the personal webpage[1].

## 1.2 Organization of the Thesis

The rest of the thesis is organized as follows:

- **Chapter 2** defines the task of speaker verification (SV) and metrics used to evaluate SV systems. It presents the evolution of SV with respect to the techniques used in the thesis, covering the i-vector approach, probabilistic linear discriminant backend, and current neural network models for speaker embedding extraction.
- **Chapter 3** provides fundamentals of theory underlying microphone array signal processing and beamforming. It defines signal models, discusses techniques we employed (such as MVDR and GEV), and mentions broadband beamforming.
- **Chapter 4** desribes datasets for single-channel speaker verification. Then, it delves into our endeavors related to defining training and evaluation datasets for multi-channel SV. It introduces our corpus, MultiSV, and its extension, MultiSV2.
- **Chapter 5** is devoted to speech separation that lies at the heart of our front-end models in cascaded multi-channel embedding extractions. It starts by reviewing approaches to single-channel and multi-channel separation/enhancement. It introduces metrics to evaluate the front-end in isolation. Then, it focuses on existing and proposed models for speech-noise separation and their evaluation.
- **Chapter 6** covers the core topic of our thesis. First, it defines multi-channel speaker verification and reviews techniques proposed in the literature. Then, it introduces our work on combating the harmful effects of reverberation. Most of the chapter is devoted to our cascaded framework for multi-channel speaker embedding extraction, its development, instances, and analysis.
- **Chapter 7** introduces and analyses our end-to-end multi-channel embedding extraction model, using foundation SSL models and a light-weight backend.
- **Chapter 8** summarizes our efforts and outlines potential future directions.

[1] https://www.fit.vut.cz/person/imosner/publications/

# Chapter 2

# Speaker Verification

Human beings possess measurable characteristics that identify individuals — known as biometrics. Their measurements include fingerprints, images of the retina or face, recordings of voice, and others. Despite being weaker (compared to fingerprints), voice is interesting from the ease-of-capture perspective. Therefore, various applications use voice biometrics for authentication (call centers), law enforcement, and personalization (home assistants, smart devices). What lies at the heart of such applications is speaker verification.

## 2.1 Task Definition

Speaker verification (SV) determines whether a pair of recordings contains the voice of the same speaker or the voices of different speakers. The pair evaluated by the SV system is called a *trial*. Conventionally, the trial comprises a recording of an *enrolled* speaker and a *test* recording. The test part of the trial contains the voice of a speaker tested for correspondence with the voice of the enrolled speaker. When the two recordings belong to the same speaker, the trial is said to be a *target* one. Otherwise, it is an *impostor* or a *non-target* trial.

Estimating whether a given pair of recordings is a target or impostor trial can be considered a binary classification problem. Therefore, four outcomes stemming from the predictor decision are possible:

- *True positive*: The target trial is correctly classified as a target trial.
- *True negative*: The impostor trial is correctly classified as an impostor trial.
- *False positive (false alarm)*: The impostor trial is mistakenly classified as a target trial.
- *False negative (miss)*: The target trial is mistakenly classified as an impostor trial.

Generally, the trial's enrollment and test parts do not need to consist of just one recording. Multiple enrollment/test utterances are also possible. This will often be the case in our experiments because we use multiple microphones that record the same utterance. However, enrollment or test recordings do not need to contain the same speech with different channel information. There are also applications where, for instance, a speaker may be enrolled with multiple utterances recorded over various channels. However, this scenario is out of the scope of this study. We will consider one enrollment and one test recording in this chapter to simplify the description.

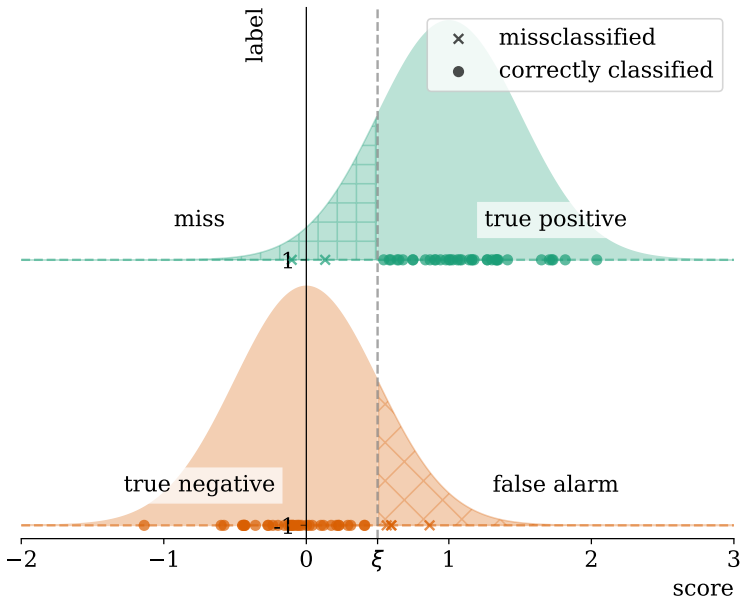



Figure 2.1: Representation of trials with their ground truth and assigned labels. Shaded areas correspond to the distributions of assigned scores. The color of the points codes the ground truth labels. The illustration considers a specific threshold $\xi$.

From the perspective of the content of the enrollment and test utterances, we can also divide speaker verification into text-dependent and text-independent. The former labels a trial as a target if and only if both word sequence and speaker identities match. On the other hand, the latter assigns a target label to a trial if speaker identities match, irrespective of the uttered message.

## 2.2 Evaluation Metrics

Speaker verification systems are evaluated on supervised sets of trials with labels marking target ($\ell = 1$) and non-target ($\ell = -1$) trials. Therefore, we can decompose the set of trials $\mathcal{P}$ into subsets of target and non-target (impostor) trials: $\mathcal{P} = \mathcal{P}_{\text{tgt}} \cup \mathcal{P}_{\text{imp}}$, where $\mathcal{P}_{\text{tgt}} = \{(p, \ell) \mid \ell = 1\}$, $\mathcal{P}_{\text{imp}} = \{(p, \ell) \mid \ell = -1\}$, and $p$ represents a trial. Given each trial from $\mathcal{P}$, the SV system provides a score (see Sections 2.4.2 and 2.4.3 for score computation), where a higher value represents a higher likelihood of the target trial prediction and vice versa. In order to obtain a binary decision, the score is compared with a threshold $\xi$. Then, trials with scores exceeding the threshold are assigned the target label. On the contrary, trials classified as non-target ones have scores lower than the threshold. Given a specific threshold, we are interested in two quantities corresponding to the system errors:

$$\begin{aligned} \text{MR}(\mathcal{P}_s, \xi) &= \frac{|\{(p, \ell, s) \mid \ell = 1 \wedge s < \xi\}|}{|\mathcal{P}_{\text{tgt}}|}, \\ \text{FAR}(\mathcal{P}_s, \xi) &= \frac{|\{(p, \ell, s) \mid \ell = -1 \wedge s > \xi\}|}{|\mathcal{P}_{\text{imp}}|}, \end{aligned} \tag{2.1}$$

where $\mathcal{P}_s$ signifies a set $\mathcal{P}$ with each trial augmented by a score $s$ (assigned by the SV system). $\mathrm{MR}(\mathcal{P}_s, \xi)$ is a *miss rate* with numerator representing the count of misses. Analogously, the numerator of $\mathrm{FAR}(\mathcal{P}_s, \xi)$ (*false alarm rate*) is the count of false alarms. In Figure 2.1, we provide an illustrative example of trials with their ground truth and assigned labels to demonstrate potential errors (introduced in the previous section) and a pictorial basis for computing the above quantities.

### 2.2.1 Detection Error Tradeoff Curve

As (2.1) suggests, the proportion of miss and false alarm rates changes as we change the threshold. Therefore, we can see a tradeoff between the two types of errors. A graph that captures this phenomenon is the Detection Error Tradeoff (DET) curve [Martin et al., 1997]. It plots the miss rate (y-axis) vs. the false alarm rate (x-axis). Moreover, its axes are non-linearly transformed with a probit transformation. It typically results in close-to-linear plots.

DET curves provide an overall view of the performance of SV systems across a wide range of operating points. Different parts of the plot might be of interest for various purposes. When the SV system is employed for authentication, a low false alarm rate region is the most interesting. On the other hand, law enforcement and other related applications require a low miss rate. Hence, the corresponding part of the DET curve is crucial when comparing systems.

### 2.2.2 Equal Error Rate

Although DET curves provide a comprehensive view of SV systems' performance, they might not be practical. It is convenient to describe the performance by a single value. One such metric is the Equal Error Rate (EER). It corresponds to one operating point on the DET curve. For such point, $\mathrm{MR}(\mathcal{P}_s, \xi)$ equals (or approximately equals[1]) $\mathrm{FAR}(\mathcal{P}_s, \xi)$. Then, EER is the value of the corresponding false alarm (or miss) rate.

Since EER compresses the performance into one value, it inevitably neglects the system's behavior in various operating regimes. Therefore, comparing different operating points for different applications might be more insightful.

### 2.2.3 Detection Cost Function

Another single-value metric, defined by the National Institute of Standards and Technology (NIST) for the ongoing series of speaker recognition (SRE) evaluations [Przybocki and Martin, 2004], is called Detection Cost Function (DCF). It focuses on an operating point of interest. It is formulated as a weighted sum of miss and false alarm error probabilities:

$$\begin{aligned} C_{\mathrm{det}}(\mathcal{P}_s, \xi) = & C_{\mathrm{miss}} P_{\mathrm{tgt}} \mathrm{MR}(\mathcal{P}_s, \xi) + \\ & C_{\mathrm{FA}} (1 - P_{\mathrm{tgt}}) \mathrm{FAR}(\mathcal{P}_s, \xi), \end{aligned} \tag{2.2}$$

where the parameters of the cost function $C_{\mathrm{miss}}$, $C_{\mathrm{FA}}$, and $P_{\mathrm{tgt}}$ are the cost of a missed detection, the cost of a false alarm, and prior probability of the target trial, respectively. NIST specified various values for the parameters throughout the evaluations, targeting specific applications. In our case, the parameters will be set as follows: $C_{\mathrm{miss}} = 1$, $C_{\mathrm{FA}} = 1$, and $P_{\mathrm{tgt}} = 0.01$.

[1] The count of trials is limited, causing steps in the DET curve; therefore, equality is not guaranteed due to the definition of miss rate and false alarm rate.

In order to improve interpretability, the cost function is normalized by $C_{\text{default}}$, which is defined as the minimum cost achievable without processing the input. It translates to always claiming that the test segment matches the target speaker or always claiming the opposite, depending on which alternative yields the lower cost:

$$\text{DCF}(\mathcal{P}_s, \xi) = \frac{C_{\text{det}}(\mathcal{P}_s, \xi)}{C_{\text{default}}}. \tag{2.3}$$

Following the intuition, the default cost is defined as:

$$C_{\text{default}} = \min \begin{cases} C_{\text{miss}} P_{\text{tgt}} \\ C_{\text{FA}}(1 - P_{\text{tgt}}) \end{cases}. \tag{2.4}$$

Given our values of the parameters, we can write:

$$C_{\text{default}} = C_{\text{miss}} P_{\text{tgt}}. \tag{2.5}$$

If the scores are well-calibrated log-likelihood ratios [Brümmer and Doddington, 2013], the threshold can be set analytically to make an optimal decision. In terms of the above-mentioned parameters, it is given as:

$$\xi = \log \left( \frac{C_{\text{FA}}}{C_{\text{miss}}} \frac{1 - P_{\text{tgt}}}{P_{\text{tgt}}} \right). \tag{2.6}$$

Given such a threshold, evaluating (2.3) yields the so-called *actual DCF*. It naturally increases when the actual scores deviate from true log-likelihood ratios as a result of misscalibration. Since calibration of scores is not our primary goal, we will not use the pre-defined threshold. Instead, we will compute a minimum detection cost by searching for the threshold that minimizes the metric:

$$\text{minDCF}(\mathcal{P}_s) = \min_{\xi} \left[ \text{DCF}(\mathcal{P}_s, \xi) \right]. \tag{2.7}$$

We note that the actual DCF of a perfectly calibrated system equals the *minimum DCF*.

## 2.3 Speaker Verification Evolution

For a long time, the field of speaker verification was resistant to the ubiquitous adoption of neural networks, and generative models were the best-performing ones. The approach of estimating the fixed-size embedding, named i-vector [Dehak et al., 2011], as a maximum a-posteriori (MAP) point estimate of a specific latent vector, discussed in Section 2.3.1, dominated both text-dependent and text-independent verification models. The i-vector extraction scheme was commonly paired with a probabilistic linear discriminant analysis (PLDA) backend [Prince and Elder, 2007] (which will be discussed in Section 2.4.2), providing verification scores. The tandem represented a long-standing state of the art. Due to the significant impact of the i-vector approach on the field and since we use it in our initial experiments, we will start by briefly presenting the basics.

### 2.3.1 The i-vector Approach

The very beginnings of the i-vector approach date back to 2008 [Burget et al., 2008]. At that time, Joint Factor Analysis (JFA) [Kenny et al., 2007] was the state-of-the-art technique.

At the Annual Frederick Jelinek Memorial Summer Workshop, investigations by the team dealing with Robust Speaker Recognition Over Varying Channels[2] were conducted, which finally led to simplification of the JFA[3]. Eventually, in the 2010s, i-vectors [Dehak et al., 2011] became a new state of the art for extracting descriptive vectors used in SV.

The i-vector approach is closely related to Gaussian mixture models (GMM). Since this text is not meant to provide a thorough description of mixture models or the training of the i-vector extractor, the basics of the GMM model will suffice for our discussion. GMM is a generative model comprising $C$ components (Gaussian probability density functions) where the space dimensionality $F$ depends on the feature vectors. The feature vector $\mathbf{x}_t \in \mathbb{R}^F$ is assumed to be generated by one of the components whose identity (represented by a hidden variable $c$) has been selected by sampling from the categorical component prior distribution $p(c)$. Formally, the GMM probability density function (reflecting marginalization over the latent variable) can thus be expressed as

$$p(\mathbf{x}_t) = \sum_{c=1}^{C} p(c)p(\mathbf{x}_t|c) = \sum_{c=1}^{C} w_c \mathcal{N}(\mathbf{x}_t; \boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c), \tag{2.8}$$

where we denote the prior $p(c)$ by $w_c$ (components weights) that sum up to one. $\boldsymbol{\Sigma}_c$ and $\boldsymbol{\mu}_c$ represent the covariance matrix and the mean of the $c$th component, respectively. Overall, the GMM model comprises parameters $\{w_c\}_{c=1}^C$, $\{\boldsymbol{\mu}_c\}_{c=1}^C$, and $\{\boldsymbol{\Sigma}_c\}_{c=1}^C$.

We would like to describe any utterance, represented by a sequence of feature vectors $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\}$, by an utterance-specific GMM. However, what hinders such a representation is the insufficiency of features for reliable GMM estimation. To facilitate the task, we will assume that a speaker- and channel-independent GMM, *universal background model* (UBM), exists, and it will serve as a base for utterance-specific models. In practice, the UBM model is trained on utterances of all available speakers. Utterance-specific GMMs are obtained by adapting the mean vectors of UBM while assuming that the covariance matrices and component weights remain unchanged (i.e., copied from the UBM). The task, hence, boils down to adapting the means of the components for each utterance. To ease the notation, it is helpful to stack component means to so-called *supervectors*:

$$\boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_1 \\ \vdots \\ \boldsymbol{\mu}_C \end{bmatrix} \in \mathbb{R}^{CF}, \quad \mathbf{u} = \begin{bmatrix} \mathbf{u}_1 \\ \vdots \\ \mathbf{u}_C \end{bmatrix} \in \mathbb{R}^{CF}. \tag{2.9}$$

We will treat the former as a random variable where samples from its distribution would correspond to an utterance-specific GMM (with $\boldsymbol{\mu}_c$ denoting the $c$th component mean vector). The latter denotes a supervector of the UBM (with $\mathbf{u}_c$ representing the analogy to $\boldsymbol{\mu}_c$). The i-vector approach assumes, that the $\boldsymbol{\mu}$ is obtained through the following transformation:

$$\boldsymbol{\mu} = \mathbf{u} + \mathbf{T}\boldsymbol{\varphi}, \tag{2.10}$$

where $\mathbf{T} \in \mathbb{R}^{CF \times I}$ is a low-rank rectangular matrix, with $I \ll CF$ denoting a subspace dimensionality. Columns of $\mathbf{T}$ represent bases spanning a subspace of the supervector space containing both speaker and session variabilities. Therefore, it is called *total variability*

[2] https://www.clsp.jhu.edu/workshops/08-workshop/robust-speaker-recognition-over-varying-channels/

[3] Since JFA is out of the scope of this document, we refer the interested reader to [Kenny et al., 2007] for more details.

*matrix* [Dehak et al., 2011]. The term $\boldsymbol{\varphi}$ represents a random variable having a standard normal prior distribution:

$$\boldsymbol{\varphi} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \tag{2.11}$$

and its components are referred to as *total factors*. Then, we can state that supervectors $\boldsymbol{\mu}$ are confined to live only in a subspace defined by $\mathbf{T}$. Therefore, their prior distribution is Gaussian:

$$\boldsymbol{\mu} \sim \mathcal{N}(\mathbf{u}, \mathbf{T}\mathbf{T}^{\intercal}). \tag{2.12}$$

Given the utterance $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\}$, we are eventually not interested in the posterior distribution of variable $\boldsymbol{\mu}$ of high dimensionality $CF$. Instead, the posterior distribution $p(\boldsymbol{\varphi}|\mathcal{X})$ of low-dimensional $\boldsymbol{\varphi}$ is of interest. The mode of that distribution, that is termed *i-vector*, is then a vector representation of the utterance. In Appendix A, we show that the posterior follows a Gaussian distribution. Therefore, the mode coincides with the mean denoted by $\boldsymbol{\varphi}_{\mathcal{X}}$.

Having equipped ourselves with the intuition behind i-vectors, let us now show how to estimate i-vectors, noting that full derivation is provided in Appendix A. Following (2.9), it will be helpful to group shared covariance matrices that are estimated during the UBM training. The block diagonal representation is given as:

$$\boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_1 & \mathbf{0} & \ldots & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_2 & \ldots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \ldots & \boldsymbol{\Sigma}_C \end{bmatrix} \in \mathbb{R}^{CF \times CF}. \tag{2.13}$$

In (2.8), we showed the probability density function of GMM. Using the Bayes rule, we can express the posterior distribution of the UBM components as

$$\gamma_{c,t} := p(c|\mathbf{x}_t) = \frac{p(\mathbf{x}_t|c)p(c)}{p(\mathbf{x}_t)} = \frac{w_c \mathcal{N}(\mathbf{x}_t; \mathbf{u}, \boldsymbol{\Sigma}_c)}{\sum_{c'=1}^{C} w_{c'} \mathcal{N}(\mathbf{x}_t; \mathbf{u}_{c'}, \boldsymbol{\Sigma}_{c'})}. \tag{2.14}$$

The term $\gamma_{c,t}$ is commonly referred to as the *occupation probability*. Given a set of $T$ feature vectors, the occupation probabilities are useful to collect Baum-Welch statistics:

$$\begin{aligned} N_c &= \sum_{t=1}^{T} \gamma_{c,t}, \\ \tilde{\mathbf{f}}_c &= \sum_{t=1}^{T} \gamma_{c,t}(\mathbf{x}_t - \mathbf{u}_c) = \mathbf{f}_c - N_c \mathbf{u}_c, \end{aligned} \tag{2.15}$$

with the former representing the zeroth- and the latter *centered* first-order statistics. Analogously to parameters, we gather the statistics in a matrix and a vector as

$$\begin{aligned} \mathbf{N} &= \begin{bmatrix} N_1\mathbf{I} & \mathbf{0} & \ldots & \mathbf{0} \\ \mathbf{0} & N_2\mathbf{I} & \ldots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \ldots & N_C\mathbf{I} \end{bmatrix} \in \mathbb{R}^{CF \times CF}, \\ \tilde{\mathbf{f}} &= \begin{bmatrix} \tilde{\mathbf{f}}_1 \\ \vdots \\ \tilde{\mathbf{f}}_C \end{bmatrix} \in \mathbb{R}^{CF}. \end{aligned} \tag{2.16}$$

Provided that the total variability matrix $\mathbf{T}$ has been estimated in advance using the expectation-maximization (EM) algorithm, the i-vector $\boldsymbol{\varphi}_{\mathcal{X}}$ of the utterance $\mathcal{X}$ can be obtained as

$$\boxed{\boldsymbol{\varphi}_{\mathcal{X}} = \boldsymbol{\Gamma}_{\boldsymbol{\varphi}_{\mathcal{X}}}^{-1} \mathbf{T}^{\intercal} \boldsymbol{\Sigma}^{-1} \tilde{\mathbf{f}},} \tag{2.17}$$

where $\boldsymbol{\Gamma}_{\boldsymbol{\varphi}_{\mathcal{X}}}$ is a precision matrix of the i-vector Gaussian posterior distribution given as

$$\boldsymbol{\Gamma}_{\boldsymbol{\varphi}_{\mathcal{X}}} = \mathbf{T}^{\intercal} \mathbf{N} \boldsymbol{\Sigma}^{-1} \mathbf{T} + \mathbf{I}. \tag{2.18}$$

We note that the provided description is minimalistic, and we refer the reader to Appendix A for a thorough description of steps leading to the expression for i-vectors.

### 2.3.2 The Era of Neural Networks

Since neural networks turned out to be a promising model in speech technology, a substantial body of literature focused on a paradigm change in speaker verification by introducing neural processing to the pipeline [Lozano-Diez et al., 2016, Bhattacharya et al., 2016].

In the traditional scheme, both i-vector and PLDA models are generative models. To promote discrimination between same-speaker and different-speaker trials right during optimization, [Burget et al., 2011] devised a discriminative training of the PLDA, resulting in performance gains. Inspired by the work, [Glembek et al., 2011] designed a discriminative training framework for the i-vector extractor parameters. The studies on discriminative training eventually led to the development of an end-to-end approach to text-independent SV, in which the individual components are pre-trained in a generative way first. They are subsequently chained and trained together with a speaker-discriminative objective [Rohdin et al., 2018, Rohdin et al., 2020].

Noteworthy progress was made in text-dependent speaker verification. [Variani et al., 2014] proposed to use a deep neural network to extract speaker features. The model was trained to discriminate between training speakers at the frame level, where the contextualized frame formed the input to the network. At test time, internal per-frame activations of the last hidden layer were aggregated, yielding a speaker embedding (*d-vector*). A cosine similarity between the test and the average of enrollment d-vectors represented a verification score. The evaluation on a proprietary dataset revealed competitive performance compared to the i-vector baseline. Building upon the success of d-vectors, [Heigold et al., 2016] designed an end-to-end model for text-dependent speaker verification. It reused the same approach to speaker embedding extraction with various alterations of the original architecture. On top of it, cosine similarity provided scores that were passed to a logistic regression jointly trained with the embedding extractor. Eventually, given test and enrollment utterances, the model directly predicted the probability of the same speaker hypothesis.

Drawing inspiration from [Burget et al., 2011] and [Heigold et al., 2016] for criterion and architecture, [Snyder et al., 2016] proposed an end-to-end model for text-independent SV. It comprised an integrated neural network that performed frame-level processing. In contrast to [Heigold et al., 2016], the frame-level part was followed by temporal pooling and a few layers, providing a speaker-descriptive embedding vector. In the last stage, the enrollment and test embeddings entered a loss function inspired by [Burget et al., 2011]. Per the evaluation results in the study, the system outperformed the i-vector baseline in EER, given a large number of training speakers. Aiming at improving the performance on small publicly available datasets and relaxing the dependence on large corpora, the study

in [Snyder et al., 2017] split the end-to-end SV pipeline into two parts. The first part relies on a DNN extracting speaker embeddings, and the second one provides verification scores given enrollment and test embeddings. The neural network of the first part comprised a frame-level stage, temporal aggregation, and a segment-level stage providing embeddings (*x-vectors*). In training, the (transformed) embeddings were mapped to speaker identities. This network laid down a basis for modern embedding extractors. Therefore, we will discuss it in detail in Section 2.4. We will also describe its important successors comprising the same high-level structure (Section 2.4.1).

## 2.4 Modern Speaker Verification

Reflecting the design suggested in [Snyder et al., 2017] and reinforcing pipeline from the era of i-vectors, current state-of-the-art single-channel approaches perform SV in two steps [Huh et al., 2023]:

1. Given a variable-length recording, a neural network extracts a speaker-descriptive vector of low (fixed) dimensionality, embedding. Therefore, such a model is commonly termed an embedding extractor. We will describe the widely adopted high-level scheme in Section 2.4.1 along with some examples of networks.

2. Embeddings extracted from the enrollment and test recordings are compared to yield a score employed to produce the final decision about the equality of identities in the trial. Various approaches to score computation have been developed or adopted from different research fields. One of the most prominent is log-likelihood ratio estimation given by a probabilistic linear discriminant analysis (PLDA) [Prince and Elder, 2007]. We will describe PLDA in Section 2.4.2. It inspired other approaches, such as the discriminative PLDA [Burget et al., 2011] mentioned above or pairwise support vector machine (PSVM) [Cumani and Laface, 2014b, Cumani and Laface, 2014a]. The most common method nowadays is cosine similarity between embeddings (Section 2.4.3). Modeling speaker embedding for score computation is still an active research field, trying to accommodate the nature of embeddings provided by recent extractors. Notable contributions in this direction are Probabilistic Spherical Discriminant Analysis (PSDA) [Brümmer et al., 2022] and its successor Toroidal Probabilistic Spherical Discriminant Analysis (T-PSDA) [Silnova et al., 2023].

### 2.4.1 Speaker Embedding Extraction

There is a broad consensus in the literature on the high-level structure of embedding extractors employing neural networks. It reflects the structure presented in [Snyder et al., 2017]:

**Frame-level processing** The most common inputs to the embedding extractors are feature vectors (e.g., log Mel-filterbank energies) extracted from speech segments, usually with silence removed by a preceding voice activity detection (VAD). In the case of time-domain input, the signal is generally pre-processed by a few layers, yielding internal vector representations with a reduced rate (compared to the initial sampling rate). Therefore, the initial part of the network still operates at a frame level. It commonly processes frames in the context of the surrounding ones. It is achieved by convolutional layers (where the receptive field increases with depth) or by Transformer encoder blocks. Pioneering architectures used 1D convolutional layers, known

as time-delay neural networks (TDNN) [Waibel et al., 1989]. Later, models employing 2D convolutions and residual connections became popular [He et al., 2016a]. With the spread of Transformers [Vaswani et al., 2017], there has been a trend to employ the attention mechanism within embedding extractors in recent years.

**Pooling** In order to extract one embedding per utterance of variable length, per-frame features need to undergo aggregation across time. The original x-vector model [Snyder et al., 2017] comprised pooling implemented as per-dimension mean and standard deviation (i.e., statistics pooling). Using the mean alone (known as temporal average pooling, TAP) appears to be a reasonable alternative. Surprisingly, [Wang et al., 2021b] showed that aggregation comprising only standard deviation pooling can yield better results than TAP, and it even outperforms statistics pooling with different backbone networks. Since not all features are equally informative concerning speaker identity (consider the difference between vowels, fricatives, or short pauses), [Zhu et al., 2018] proposed a reasonable extension of statistics pooling by introducing an attention mechanism to provide weights for all features and their elements. Then, the mean and standard deviation of weighted per-frame features replaced the standard statistics pooling. Concerned with the fact that the methods mentioned above pool information over time for each feature dimension, irrespective of other dimensions, [Stafylakis et al., 2021] took a different approach. They devised a method where frequency-dependent channel-wise correlations replaced per-dimension statistics. An alternative, considering 2D (instead of 3D) representations, was successfully introduced in [Stafylakis et al., 2023]. The other two approaches we will mention explore a different strategy for expressing variable-length input by a fixed-sized vector. NetVLAD [Arandjelovic et al., 2016] and its successor GhostVLAD [Zhong et al., 2019] extended the "vector of locally aggregated descriptors" (VLAD) method known in computer vision by allowing differentiability to be compatible with neural networks. Compression into a fixed-sized embedding is achieved via the notion of clusters, centroids, and the expression of residuals with respect to them. [Xie et al., 2019] presented a successful adoption in speaker embedding extraction. Finally, given a specific type of architecture performing the per-frame processing, the pooling method can be tailored specifically to it. Multi-head factorized attention pooling is an example of such aggregation [Peng et al., 2023a]. We will discuss it in detail later in this section, as we will employ it in our experiments.

**Utterance-level processing** Utterance-level processing aims to turn aggregated information into the speaker embedding and facilitate network training. The most straightforward yet common approach projects the output of the pooling layer via a linear (or affine) transformation to the speaker embedding. An alternative is to use multiple affine transformations with non-linearities in between [Snyder et al., 2017]. The embedding can be retrieved at various points. Finally, an affine transformation projects the embedding or the final internal representation to per-speaker logits used during training.

### Extractors Using Time-Delay Neural Networks

A breakthrough architecture for speaker embedding extraction with neural networks, shown in Figure 2.2, has de facto changed the paradigm in the speaker verification field [Snyder et al., 2017, Snyder et al., 2018]. The network used 20-dimensional Mel-frequency cepstral

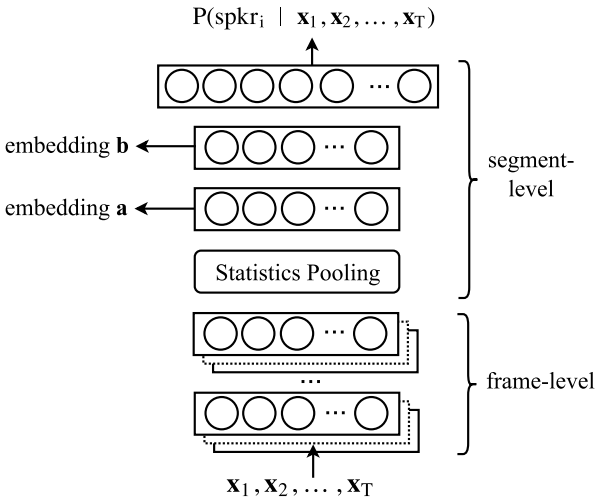



Figure 2.2: Architecture of the original x-vector extractor. In the training phase, it optimizes the posterior probability of the speaker given a variable number of input features. During evaluation time, either embedding **a** or **b** or both are extracted to obtain a fixed-length representation of an utterance. Reprinted from [Snyder et al., 2017].

coefficients (MFCCs) with a frame length of 25 ms. The frame-level part comprised layers with a so-called time-delay neural network (TDNN) architecture [Peddinti et al., 2015]. They are equal to 1D convolutional layers. Focusing on the $t$th frame, the first layer considered the context $\{t\text{-}2, t\text{-}1, t, t{+}1, t{+}2\}$. The following two focused on frames $\{t\text{-}2, t, t{+}2\}$ and $\{t\text{-}3, t, t{+}3\}$, respectively (which resembles a dilated convolution). The last two frame-level layers operated without any context. The ReLU activation function was used as a non-linearity between layers. Frame-level representations of the last layer were subject to statistics (mean and standard deviation) pooling. Finally, two utterance-level layers processed aggregated statistics. Activations of both of them were considered speaker embeddings (termed embedding **a** and **b**) in [Snyder et al., 2017]. The former turned out better and coined the widely adopted name *x-vector* [Snyder et al., 2018]. Eventually, a classification layer (for speaker classes) with the softmax function was appended to the described network. The final model comprising 4.4 M parameters was trained to optimize the standard cross-entropy.

Various flavors of the original TDNN x-vector architecture emerged over the years [Villalba et al., 2020]. A notable version is the one using factorized TDNN layers [Povey et al., 2018]. A successful alternative was the extended TDNN (E-TDNN) [Snyder et al., 2019], representing a deeper network with a longer context (11 frames on each side of the center frame) and interleaved architecture, where per-frame operating layers followed layers with context (dilated convolutions) as shown in Table 2.1. E-TDNN is the x-vector architecture version used in our experiments (among other embedding extractors). Finally, a prominent architecture that became the baseline for many studies is ECAPA-TDNN [Desplanques et al., 2020]. It benefits from recent advances, such as Res2Net blocks [Gao et al., 2021], squeeze-and-excitation [Hu et al., 2020a], or channel- and context-dependent statistics pooling.

Table 2.1: The extended x-vector topology proposed in [Snyder et al., 2019]. The network takes 30-dimensional MFCC features as inputs; $S$ is the number of training speakers.

| **Layer** | **Layer context** | **(Input) × output** |
|---|---|---|
| frame1 | $[t-2, t-1, t, t+1, t+2]$ | $(5 \times 30) \times 512$ |
| frame2 | $[t]$ | $512 \times 512$ |
| frame3 | $[t-2, t, t+2]$ | $(3 \times 512) \times 512$ |
| frame4 | $[t]$ | $512 \times 512$ |
| frame5 | $[t-3, t, t+3]$ | $(3 \times 512) \times 512$ |
| frame6 | $[t]$ | $512 \times 512$ |
| frame7 | $[t-4, t, t+4]$ | $(3 \times 512) \times 512$ |
| frame8 | $[t]$ | $512 \times 512$ |
| frame9 | $[t]$ | $512 \times 1500$ |
| stats pooling | $[0, T)$ | $1500 \times 3000$ |
| segment1 | 0 | $3000 \times 512$ |
| segment2 | 0 | $512 \times 512$ |
| softmax | 0 | $512 \times S$ |

### Extractors Using Residual Networks

The speaker verification field quickly adopted an influential neural network architecture designed in the context of computer vision, residual network (ResNet) [He et al., 2016a]. As the name suggests, the key concept of ResNets is residual connections. They ease optimization and allow for deep architectures by tackling the vanishing gradient problem. Consequently, ResNets achieve considerably large receptive fields by stacking multiple convolutional layers.

ResNet-based speaker embedding extractors turned out to be instrumental in achieving the best-performing system in the VoxCeleb speaker recognition challenge (VoxSRC) 2019 [Chung et al., 2019]. Multiple teams also explored them in the 2019 NIST Audio-Visual Speaker Recognition Evaluation [Sadjadi et al., 2020], where their utility became apparent. Since then, various versions have been devised [He et al., 2016b, Xie et al., 2017, Gao et al., 2021]. ResNets still represent competitive models, especially when using deep architectures (e.g., ResNet293 [Chen et al., 2023]), massive pre-training corpora, such as VoxBlink 1, 2 [Lin et al., 2024b, Lin et al., 2024a], or large-margin fine-tuning [Thienpondt et al., 2021].

Inspired by their success, we employed ResNets in our experiments. The base architecture is presented in Table 2.2. Log Mel-filterbank energy (*fbank*) feature vectors (after voice activity detection, VAD) enter the first 2D convolutional layer with a kernel size of 3×3, increasing the number of channels from 1 to 64. The output is passed to a sequence of stages, where each stage comprises multiple residual blocks that operate on representations of a fixed dimensionality. The first block in each stage is responsible for decreasing the dimensionality of features through strided convolution and increasing the number of channels (except for the very first stage, which does not alter dimensions). Outputs of the last block of the last stage are subject to flattening (collapsing feature and channel dimensions to one dimension) and statistics pooling. In order to obtain the speaker embeddings, an affine transformation projects statistics to a lower-dimensional space. The resulting embeddings are then used as the input for the classification layer. This basic processing is common to the architectures we used. Specific deviations will be described in Sections 6.4.4 and 6.4.5.

Table 2.2: ResNet34 architecture used in our experiments. $T$ represents the number of input frames and $S$ the number of training speakers. Each residual block in square brackets comprises two 2D convolutional layers, where the notation is as follows: kernel size, the number of output channels. The output size is without the channel dimension.

| **Layer name** | **Output size** | **Building block** |
|---|---|---|
| Input | $T \times 80$ | — |
| Conv 2D | $T \times 80$ | $3 \times 3$, 64 |
| Stage 1 | $T \times 80$ | $\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 3$ |
| Stage 2 | $T/2 \times 40$ | $\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 4$ |
| Stage 3 | $T/4 \times 20$ | $\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 6$ |
| Stage 4 | $T/8 \times 10$ | $\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 3$ |
| Reshaping | $T/8 \times 2560$ | — |
| Aggregation | 5120 | statistics pooling |
| Embedding | 256 | affine transformation |
| Classification head | $S$ | affine transformation |

#### Extractors using Transformer Encoder Blocks

With the advent and spread of self-attention-based architectures [Vaswani et al., 2017], studies employing Transformer encoder blocks for frame-level processing have emerged [Safari et al., 2020]. Despite benefits, such as a small number of parameters, they were not convincing enough to be widely adopted. However, the situation has changed after the release of general-purpose speech foundation models, such as wav2vec 2.0 [Baevski et al., 2020], Hubert [Hsu et al., 2021], or WavLM [Chen et al., 2022a]. Their architectures comprise a convolutional encoder followed by a stack of Transformer encoder blocks. They aim to learn speech representations via self-supervised learning (SSL) by optimizing contrastive [Baevski et al., 2020] or masked prediction [Hsu et al., 2021, Chen et al., 2022a] loss on massive speech corpora. Such representations are suitable for various speech processing tasks (including recognition, detection, semantics, speaker-, and paralinguistics-related tasks), as demonstrated by the results obtained from the SUPERB benchmark [Yang et al., 2021].

Since speech features provided by the foundation models lead to new state-of-the-art results in various tasks, interest in them has also substantially increased in the SV community. The studies can be roughly divided into those that 1) used foundation models to provide (trainable) features passed to a downstream model, and 2) used them as a frame-level processing part of embedding extraction, followed by some pooling function.

Falling within the former group, [Novoselov et al., 2023] designed an embedding extractor comprising the wav2vec 2.0 model followed by a TDNN-based embedding extractor. Instead of taking the last layer output, the SUPERB benchmark defined that a learnable weighted sum of per-layer representations of the foundation model should be used as input to the downstream model [Yang et al., 2021]. For the task of SV, the SUPERB benchmark study designed a model where the weighted sum of features enters the TDNN speaker em-

bedding model. Along the same lines, [Chen et al., 2022b] replaced TDNN by its successor, ECAPA-TDNN [Desplanques et al., 2020].

The latter group of works fits the described structure of general embedding extractors more intuitively, as the foundation model can be perceived as the frame-processing logical block, followed by pooling and utterance-level processing. [Fan et al., 2021] took the most straightforward way. They directly passed the output of the wav2vec 2.0 model to the TAP pooling. A more sophisticated approach to pooling explicitly tailored for the task of speaker embedding extraction while considering phonetic information learned by the foundation model is called multi-head factorized attention (MHFA) pooling [Peng et al., 2023a]. Since we employ it in our experiments, we will dedicate the following section to it.

### Multi-Head Factorized Attention Pooling

Using foundation models as (potentially trainable) feature extractors for well-established speaker embedding extractors seems wasteful regarding the resources and potential of such representations. Multi-head factorized attention (MHFA) pooling [Peng et al., 2023a] attempted to tackle the abovementioned issues. It represents a lightweight model compared to standard embedding extractors. Nevertheless, it empirically performs competitively. Arguably, it is due to the principled design that considers properties of speech features discussed in [Hsu et al., 2021, Chen et al., 2022a]: Patterns extracted from speech by the initial layers of foundation models are informative of speaker characteristics. It seems to be important for the SSL-trained networks to eventually suppress speaker information in later layers, as it is likely unimportant for the mask prediction loss. At the same time, the final layers carry phonetic information to enable the prediction of masked tokens from surrounding ones. In the context of embedding extraction, it might be important to focus on the initial layers conditioned on the utterance content because different phonemes (or sequences of phonemes) might be differently informative regarding speaker identity. The design of MHFA reflects this rationale.

MHFA is theoretically capable of grouping frame-level representations according to acoustic units identified by the foundation model. Representations in such groups can be pooled and combined to yield the speaker embedding. We should note that the method relies on the capacity of foundation models, as no supervision is used to enforce the described grouping behavior

Compared to the SUPERB benchmark, which combines layer representations of the foundation model via weighted average, MHFA uses two sets of weights as shown in Figure 2.3: $\mathbf{w}^{(V)}$ and $\mathbf{w}^{(K)} \in \mathbb{R}^L$. The former weighs $L$ layers to provide *values* that contain speaker information for the attention mechanism. The latter should emphasize layers with the phonetic information eventually encoded in *keys*. Denoting per-layer outputs[4] of the foundation model as $\mathbf{H}^{(l)} \in \mathbb{R}^{\mathfrak{T}\times\mathfrak{D}}$, $l = 1, \ldots, L$, the values and keys are given as:

$$\begin{aligned} \mathbf{V} &= \left(\sum_{l=1}^{L} w_l^{(V)} \mathbf{H}^{(l)}\right) \mathbf{W}^{(V)}, \\ \mathbf{K} &= \left(\sum_{l=1}^{L} w_l^{(K)} \mathbf{H}^{(l)}\right) \mathbf{W}^{(K)}, \end{aligned} \tag{2.19}$$

[4]Please note that we use the symbol $T$ when referring to the number of frames obtained by pre-defined operations (such as feature extraction). The symbol $\mathfrak{T}$ denotes the number of frames obtained via trainable neural network encoders.

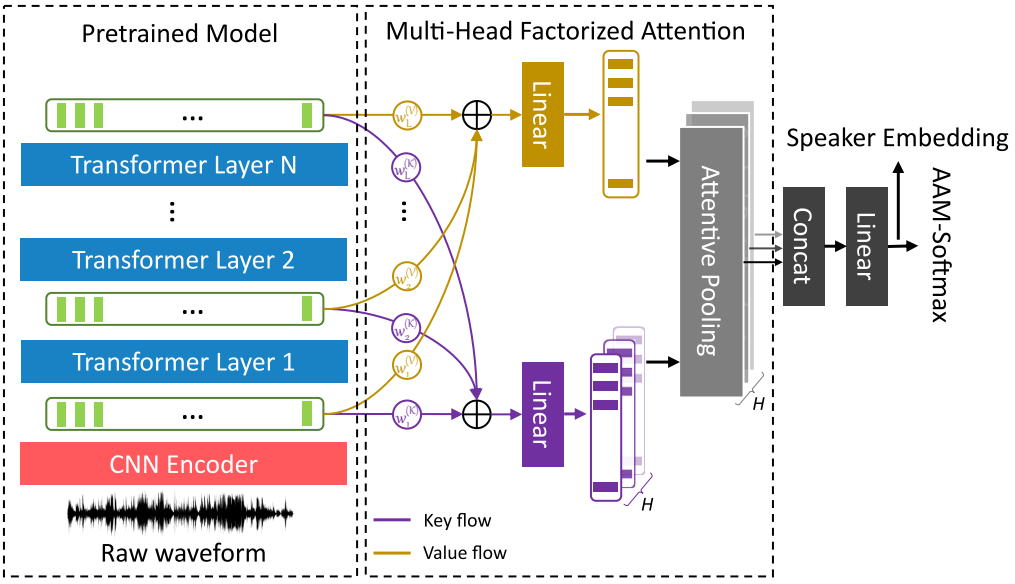



Figure 2.3: Convolution-free multi-head factorized attention pooling scheme (middle) with the foundation model (left) and subsequent embedding extraction and classification head (right). Reprinted from [Peng et al., 2023a].

with $\mathbf{W}^{(V)} \in \mathbb{R}^{\mathfrak{D} \times D}$ and $\mathbf{W}^{(K)} \in \mathbb{R}^{\mathfrak{D} \times D}$ representing projection matrices for dimensionality reduction. MHFA incorporates a trainable query matrix $\mathbf{Q} \in \mathbb{R}^{H \times D}$, where $H$ represents so-called heads. Please note that heads in MHFA do not directly correspond to those in multi-headed attention [Vaswani et al., 2017]. It is assumed that each head corresponds to one or more acoustic units. Therefore, comparing heads with phonetically aware keys should yield attention weights emphasizing given acoustic units. The obtained attention weights are applied to speaker-related values via:

$$\mathbf{C} = \text{softmax}(\mathbf{Q}\mathbf{K}^{\intercal})\mathbf{V} \in \mathbb{R}^{H \times D}, \tag{2.20}$$

where softmax is applied over columns. Eventually, the aggregated representation $\mathbf{C}$ is serialized to the utterance-level vector $\mathbf{c} \in \mathbb{R}^{HD}$, corresponding to the output of the pooling operation.

### Training Objective Functions

The original x-vector extractor [Snyder et al., 2017] was trained using a multiclass cross-entropy objective. A combination of softmax activation function and cross-entropy is usually referred to as *softmax loss*:

$$\mathcal{L}_{\text{softmax}}(\boldsymbol{\varphi}_i, \ell_i) = -\log \frac{\exp(\mathbf{w}_{\ell_i}^{\intercal} \boldsymbol{\varphi}_i + b_{\ell_i})}{\sum_j \exp(\mathbf{w}_j^{\intercal} \boldsymbol{\varphi}_i + b_j)}. \tag{2.21}$$

The term $\boldsymbol{\varphi}_i \in \mathbb{R}^I$ is an $I$-dimensional embedding of the $i$th example in a mini-batch, and the speaker identity of the segment is denoted by the label index $\ell_i$. $\mathbf{w}_j \in \mathbb{R}^I$ is the $j$th column of the weight matrix $\mathbf{W} \in \mathbb{R}^{I \times S}$. The $j$th element of a bias vector is denoted as $b_j$. $S$ represents the number of speakers in a training set. The softmax loss does not explicitly enforce intra-class compactness and inter-class separability of embeddings, which

is useful/required for (not only speaker) verification tasks. Therefore, various types of loss functions have been developed to enforce the mentioned favorable properties of embeddings.

One of the modifications of the softmax loss that explicitly attempts to separate embeddings by angular margin controlled by the hyperparameter $m$ is additive margin softmax (*AM-softmax*) loss [Wang et al., 2018]. It is defined as follows:

$$\mathcal{L}_{\text{AM-softmax}}(\boldsymbol{\varphi}_i, \ell_i) = -\log \frac{\exp[s(\cos(\psi_{\ell_i,i}) - m)]}{\exp[s(\cos(\psi_{\ell_i,i}) - m)] + \sum_{j \neq \ell_i} \exp[s\cos(\psi_{j,i})]}, \tag{2.22}$$

where

$$\cos(\psi_{\ell_i,i}) = \frac{\mathbf{w}_{\ell_i}^\intercal \boldsymbol{\varphi}_i}{\|\mathbf{w}_{\ell_i}\|_2 \|\boldsymbol{\varphi}_i\|_2}, \tag{2.23}$$

with $\psi_{\ell_i,i}$ representing the angle between the prototype $\mathbf{w}_{\ell_i}$ and the embedding $\boldsymbol{\varphi}_i$. The scaling factor $s$ is a hyperparameter of the loss function. It was introduced to ensure gradients of sufficient magnitude.

Another alternative, additive angular margin softmax (*AAM-softmax*) loss [Deng et al., 2019] explicitly adds a margin to angles (as opposed to AM-softmax). The loss function is defined as

$$\mathcal{L}_{\text{AAM-softmax}}(\boldsymbol{\varphi}_i, \ell_i) = -\log \frac{\exp[s\cos(\psi_{\ell_i,i} + m)]}{\exp[s\cos(\psi_{\ell_i,i} + m)] + \sum_{j \neq \ell_i} \exp[s\cos(\psi_{j,i})]}. \tag{2.24}$$

Although multiple other loss functions have been either devised for speaker embedding extractors or adopted from other research fields, we mentioned just the three training criteria because we employed them in our experiments.

### 2.4.2 Probabilistic Linear Discriminant Analysis

Probabilistic linear discriminant analysis (PLDA) is a generative model commonly used as a backend for SV systems. It models the distribution of speaker embeddings (be they i-vectors or neural network representations), assuming two sources of variability. The first relates to variability between individuals. The second one concerns within-individual variability. The original model was proposed in the context of face recognition [Prince and Elder, 2007]. As it turned out to be promising to model individuals' embeddings, it soon attracted attention in the SV community [Kenny, 2010]. As summarized in [Sizov et al., 2014], the SV field has explored various flavours of the PLDA models. We will limit our discussion to the original model for context and its simplified version, which we used in our experiments.

#### Standard PLDA

We will begin by describing the original model [Prince and Elder, 2007] that will be referred to as *standard PLDA*. Later, we will briefly mention its alternatives, eventually providing more details about the specific one used in some of our experiments. In the standard PLDA[5], the embedding $\boldsymbol{\varphi}_{s,u} \in \mathbb{R}^I$ of the $u$th utterance of the speaker $s$ is modeled as

$$\boldsymbol{\varphi}_{s,u} = \mathbf{m} + \mathbf{U}\mathbf{y}_s + \mathbf{V}\mathbf{x}_{s,u} + \boldsymbol{\epsilon}_{s,u}, \tag{2.25}$$

where $\mathbf{m}$ is a mean vector of embeddings, $\mathbf{U} \in \mathbb{R}^{I \times D_1}$ is a loading matrix defining a subspace of the embedding space characterizing speaker variability. $\mathbf{V} \in \mathbb{R}^{I \times D_2}$ is a loading

[5]Despite the original paper dealing with face recognition, we intentionally use SV-related terms not to clutter the text with terms from computer vision.

matrix whose columns define a subspace of channel variability, with $D_1 \leq I$ and $D_2 \leq I$. We note that embeddings are usually centered in practice, making $\mathbf{m}$ a zero vector. The hidden variables $\mathbf{y}_s \in \mathbb{R}^{D_1}$, $\mathbf{x}_{s,u} \in \mathbb{R}^{D_2}$ (which can be viewed as factor loadings) represent the inter-speaker and intra-speaker variability (often called *channel*), respectively. The variable $\boldsymbol{\epsilon}_{s,u}$ denotes residual noise, i.e., remaining unexplained variation in the data. In the standard PLDA, a normal prior is imposed on all the variables:

$$\mathbf{y}_s \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \tag{2.26}$$

$$\mathbf{x}_{s,u} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \tag{2.27}$$

$$\boldsymbol{\epsilon}_{s,u} \sim \mathcal{N}(\mathbf{0}, \mathbf{D}^{-1}), \tag{2.28}$$

with $\mathbf{D}$ representing a diagonal precision matrix of a residual noise. The description above gave us insight into the generative process of speaker embeddings. Verbatim, we can summarize it as follows: First, a vector $\mathbf{y}_s$ is sampled from its prior distribution, providing the speaker's identity. Then it is projected to an embedding space. We note that if $\mathbf{U}$ is a rectangular low-rank matrix, the speaker representations are confined to live in a subspace of the embedding space. The channel-specific variable $\mathbf{x}_{s,u}$ is also sampled from its prior distribution and projected to the embedding space. The same conclusions about the subspace it lives in hold here as well. Finally, a residual noise variable is drawn from a normal distribution with precision $\mathbf{D}$. The sum of the components with the mean eventually provides the observed embedding $\boldsymbol{\varphi}_{s,u}$.

Given the model definition and the speaker identity $\mathbf{y}_s$, a conditional distribution of embeddings is

$$\boldsymbol{\varphi}_{s,u}|\mathbf{y}_s \sim \mathcal{N}(\mathbf{m} + \mathbf{U}\mathbf{y}_s, \mathbf{V}\mathbf{V}^\intercal + \mathbf{D}^{-1}). \tag{2.29}$$

We can further identify distribution of speaker-specific vectors $\tilde{\mathbf{y}}_s$ in the embedding space giving

$$\begin{aligned} \tilde{\mathbf{y}}_s &\sim \mathcal{N}(\mathbf{m}, \boldsymbol{\Sigma}_{ac}), \\ \boldsymbol{\varphi}_{s,u}|\tilde{\mathbf{y}}_s &\sim \mathcal{N}(\tilde{\mathbf{y}}_s, \boldsymbol{\Sigma}_{wc}), \end{aligned} \tag{2.30}$$

where $\boldsymbol{\Sigma}_{ac} = \mathbf{U}\mathbf{U}^\intercal$ represents the across-class covariance matrix and $\boldsymbol{\Sigma}_{wc} = \mathbf{V}\mathbf{V}^\intercal + \mathbf{D}^{-1}$ is the within-class covariance matrix. It is the way of representing the two covariance matrices that differentiates variants of the PLDA. In the *simplified PLDA* [Kenny, 2010] (which will be described in detail later), the across-class covariance is also $\mathbf{U}\mathbf{U}^\intercal$, but the within-class covariance is represented by one full matrix. In the *two-covariance model*, as the name indicates, two full covariance matrices are used. All the mentioned alternatives impose Gaussian priors on the factors. A variant employing Student's t-distribution priors was introduced in [Kenny, 2010]. Another version of that model was presented in [Brümmer et al., 2018]. Due to the use of the specific heavy-tailed priors, they are usually referred to as *heavy-tailed PLDA* (HT-PLDA).

An important question is how to train the PLDA model (i.e., matrices $\mathbf{U}$, $\mathbf{V}$, and $\mathbf{D}$) given embeddings and their labels. Since the model is not a primary focus of our work and the training is rather involved, we will not dive into it. We note that the EM algorithm used for maximum likelihood training is thoroughly described in [Brümmer, 2010a].

From the perspective of experiments, it is helpful to concentrate on how to use the PLDA model in the context of SV and, specifically, obtain a score for a pair of embeddings[6]. In

[6]Even though it is possible to consider an arbitrary number of enrollment and test embeddings, we will limit our discussion to the case of one embedding per enrollment and one per test. After all, this is always the case in our experiments since we focus on fusing information from different microphones at the acoustic level.

the PLDA framework, score computation can be framed as comparing the likelihoods of the data (embeddings) under different hypotheses (models). Each hypothesis corresponds to a different relationship between latent variables and the observation. The first, the same-speaker hypothesis ($\mathcal{H}_0$), assumes that one speaker was responsible for uttering two given speech segments (represented by embeddings). In other words, the embeddings share the same underlying latent variable, $\mathbf{y}_{12}$, while being differentiated by different channel latent variables $\mathbf{x}_1$ and $\mathbf{x}_2$. The different-speaker hypothesis ($\mathcal{H}_1$) supposes that the two speech segments correspond to two different speaker identities, i.e., speaker-specific latent variables $\mathbf{y}_1$ and $\mathbf{y}_2$ are not shared. Mathematically, we can express the comparison of hypotheses as the log-likelihood ratio (LLR):

$$s(\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2) = \log \frac{p(\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2|\mathcal{H}_0)}{p(\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2|\mathcal{H}_1)}. \tag{2.31}$$

Since speaker-specific latents are assumed independent under the $\mathcal{H}_1$ hypothesis, the verification score can be formulated as

$$s(\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2) = \log \frac{p(\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2|\mathcal{H}_0)}{p(\boldsymbol{\varphi}_1|\mathcal{H}_1)p(\boldsymbol{\varphi}_2|\mathcal{H}_1)}. \tag{2.32}$$

Respecting the models given by hypotheses and PLDA, the likelihoods in (2.32) can be expressed via the following marginalizations:

$$\begin{aligned} p(\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2|\mathcal{H}_0) &= \int\int\int p(\boldsymbol{\varphi}_1|\mathbf{y}_{12}, \mathbf{x}_1)p(\boldsymbol{\varphi}_2|\mathbf{y}_{12}, \mathbf{x}_2)p(\mathbf{y}_{12})p(\mathbf{x}_1)p(\mathbf{x}_2)d\mathbf{y}_{12}d\mathbf{x}_1d\mathbf{x}_2, \\ p(\boldsymbol{\varphi}_1|\mathcal{H}_1) &= \int\int p(\boldsymbol{\varphi}_1|\mathbf{y}_1, \mathbf{x}_1)p(\mathbf{y}_1)p(\mathbf{x}_1)d\mathbf{y}_1d\mathbf{x}_1, \end{aligned} \tag{2.33}$$

where $p(\boldsymbol{\varphi}_2|\mathcal{H}_1)$ would be expressed analogically. As it follows from the PLDA model definition, $p(\boldsymbol{\varphi}_1|\mathbf{y}_{12}, \mathbf{x}_1)$ is also Gaussian distribution. Instead of explicitly solving integrals, we choose to express the LLR similarly to [Prince and Elder, 2007]. Combining generative equations that lead to expressing observed embeddings under the two hypotheses, we get:

$$\begin{aligned} \mathcal{H}_0 : \begin{bmatrix}\boldsymbol{\varphi}_1 \\ \boldsymbol{\varphi}_2\end{bmatrix} &= \begin{bmatrix}\mathbf{m} \\ \mathbf{m}\end{bmatrix} + \begin{bmatrix}\mathbf{U} & \mathbf{V} & \mathbf{0} \\ \mathbf{U} & \mathbf{0} & \mathbf{V}\end{bmatrix} \begin{bmatrix}\mathbf{y}_{12} \\ \mathbf{x}_1 \\ \mathbf{x}_2\end{bmatrix} + \begin{bmatrix}\boldsymbol{\epsilon}_1 \\ \boldsymbol{\epsilon}_2\end{bmatrix}, \\ \mathcal{H}_1 : \begin{bmatrix}\boldsymbol{\varphi}_1 \\ \boldsymbol{\varphi}_2\end{bmatrix} &= \begin{bmatrix}\mathbf{m} \\ \mathbf{m}\end{bmatrix} + \begin{bmatrix}\mathbf{U} & \mathbf{0} & \mathbf{V} & \mathbf{0} \\ \mathbf{0} & \mathbf{U} & \mathbf{0} & \mathbf{V}\end{bmatrix} \begin{bmatrix}\mathbf{y}_1 \\ \mathbf{y}_2 \\ \mathbf{x}_1 \\ \mathbf{x}_2\end{bmatrix} + \begin{bmatrix}\boldsymbol{\epsilon}_1 \\ \boldsymbol{\epsilon}_2\end{bmatrix}. \end{aligned} \tag{2.34}$$

Rewriting the expressions in a more compact way by assigning names to vectors and matrices, we obtain

$$\mathcal{H}_0 : \underline{\boldsymbol{\varphi}} = \underline{\mathbf{m}} + \underline{\mathbf{S}}_0\underline{\mathbf{y}}_0 + \underline{\boldsymbol{\epsilon}}, \qquad \mathcal{H}_1 : \underline{\boldsymbol{\varphi}} = \underline{\mathbf{m}} + \underline{\mathbf{S}}_1\underline{\mathbf{y}}_1 + \underline{\boldsymbol{\epsilon}}, \tag{2.35}$$

where we can realize that the prior distributions of $\underline{\mathbf{y}}_0$ and $\underline{\mathbf{y}}_1$ are standard normal. Then the distribution of $\underline{\boldsymbol{\varphi}}$ (corresponding to a joint distribution of $\boldsymbol{\varphi}_1$ and $\boldsymbol{\varphi}_2$) under the $\mathcal{H}_0$ hypothesis is $\underline{\boldsymbol{\varphi}}|\mathcal{H}_0 \sim \mathcal{N}(\underline{\mathbf{m}}, \underline{\mathbf{S}}_0\underline{\mathbf{S}}_0^\intercal + \underline{\mathbf{D}}^{-1})$ and under the $\mathcal{H}_1$ hypothesis, it is $\underline{\boldsymbol{\varphi}}|\mathcal{H}_1 \sim \mathcal{N}(\underline{\mathbf{m}}, \underline{\mathbf{S}}_1\underline{\mathbf{S}}_1^\intercal + \underline{\mathbf{D}}^{-1})$, where

$$\underline{\mathbf{D}}^{-1} = \begin{bmatrix}\mathbf{D}^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{D}^{-1}\end{bmatrix}. \tag{2.36}$$

Using the provided definitions in (2.32), the verification score can be rewritten as:

$$\begin{aligned} s(\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2) = &\log \mathcal{N}\left(\begin{bmatrix}\boldsymbol{\varphi}_1 \\ \boldsymbol{\varphi}_2\end{bmatrix}; \begin{bmatrix}\mathbf{m} \\ \mathbf{m}\end{bmatrix}, \begin{bmatrix}\mathbf{U}\mathbf{U}^\intercal + \mathbf{V}\mathbf{V}^\intercal & \mathbf{U}\mathbf{U}^\intercal \\ \mathbf{U}\mathbf{U}^\intercal & \mathbf{U}\mathbf{U}^\intercal + \mathbf{V}\mathbf{V}^\intercal\end{bmatrix} + \underline{\mathbf{D}}^{-1}\right) - \\ &\log \mathcal{N}\left(\begin{bmatrix}\boldsymbol{\varphi}_1 \\ \boldsymbol{\varphi}_2\end{bmatrix}; \begin{bmatrix}\mathbf{m} \\ \mathbf{m}\end{bmatrix}, \begin{bmatrix}\mathbf{U}\mathbf{U}^\intercal + \mathbf{V}\mathbf{V}^\intercal & \mathbf{0} \\ \mathbf{0} & \mathbf{U}\mathbf{U}^\intercal + \mathbf{V}\mathbf{V}^\intercal\end{bmatrix} + \underline{\mathbf{D}}^{-1}\right). \end{aligned} \tag{2.37}$$

Using the definition of across-class and within-class covariance, we can reformulate it as:

$$\begin{aligned} s(\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2) = &\log \mathcal{N}\left(\begin{bmatrix}\boldsymbol{\varphi}_1 \\ \boldsymbol{\varphi}_2\end{bmatrix}; \begin{bmatrix}\mathbf{m} \\ \mathbf{m}\end{bmatrix}, \begin{bmatrix}\boldsymbol{\Sigma}_{tot} & \boldsymbol{\Sigma}_{ac} \\ \boldsymbol{\Sigma}_{ac} & \boldsymbol{\Sigma}_{tot}\end{bmatrix}\right) - \\ &\log \mathcal{N}\left(\begin{bmatrix}\boldsymbol{\varphi}_1 \\ \boldsymbol{\varphi}_2\end{bmatrix}; \begin{bmatrix}\mathbf{m} \\ \mathbf{m}\end{bmatrix}, \begin{bmatrix}\boldsymbol{\Sigma}_{tot} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_{tot}\end{bmatrix}\right), \end{aligned} \tag{2.38}$$

where the total variability covariance matrix $\boldsymbol{\Sigma}_{tot} = \boldsymbol{\Sigma}_{ac} + \boldsymbol{\Sigma}_{wc}$. This expression is general and holds for different variants of PLDA. The difference is just in the definition of $\boldsymbol{\Sigma}_{ac}$ and $\boldsymbol{\Sigma}_{wc}$.

### Simplified PLDA

As indicated above, the simplified PLDA [Kenny, 2010] is very similar to the standard PLDA. The key difference resides in handling channel variability. Formally, the embedding of the utterance $u$ by speaker $s$ is modeled as:

$$\boldsymbol{\varphi}_{s,u} = \mathbf{m} + \mathbf{U}\mathbf{y}_s + \boldsymbol{\epsilon}_{s,u}. \tag{2.39}$$

Compared to the standard PLDA in [Prince and Elder, 2007], there is no explicit channel-specific latent variable with a corresponding loading matrix. Instead, channel variability is entirely described by $\boldsymbol{\epsilon}_{s,u}$ directly in the embedding space. Other terms correspond to those in (2.25). Prior distributions of variables are defined as follows:

$$\mathbf{y}_s \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \tag{2.40}$$

$$\boldsymbol{\epsilon}_{s,u} \sim \mathcal{N}(\mathbf{0}, \mathbf{E}^{-1}), \tag{2.41}$$

where $\mathbf{E} \in \mathbb{R}^{I \times I}$ is a full precision matrix of noise. Knowing the priors, we can explicitly write speaker- and noise-specific distributions in the embedding space similarly as in (2.30):

$$\begin{aligned} \tilde{\mathbf{y}}_s &\sim \mathcal{N}(\mathbf{m}, \boldsymbol{\Sigma}_{ac}), \\ \boldsymbol{\varphi}_{s,u} | \tilde{\mathbf{y}}_s &\sim \mathcal{N}(\tilde{\mathbf{y}}_s, \boldsymbol{\Sigma}_{wc}), \end{aligned} \tag{2.42}$$

where the across-class covariance matrix $\boldsymbol{\Sigma}_{ac} = \mathbf{U}\mathbf{U}^\intercal$ is the same as in the standard PLDA. In contrast, the within-class covariance matrix is directly $\boldsymbol{\Sigma}_{wc} = \mathbf{E}^{-1}$.

While the details on the training of the model parameters can be found in [Brümmer, 2010b], we will again concentrate on the score computation for a pair of embeddings. Drawing inspiration from combining generative equations for embeddings under the two hypotheses presented in the context of the standard PLDA, we can directly use the result in (2.38) and continue to explicitly express the score. For completeness, we provide the generative equations in Appendix B.1. Naming the covariance matrices in (2.38) as $\boldsymbol{\Sigma}_0$ and $\boldsymbol{\Sigma}_1$, we can explicitly express the Gaussian density functions:

$$\begin{aligned} s(\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2) = &-\frac{1}{2}\log|\boldsymbol{\Sigma}_0| - \frac{1}{2}\left(\begin{bmatrix}\boldsymbol{\varphi}_1 \\ \boldsymbol{\varphi}_2\end{bmatrix} - \begin{bmatrix}\mathbf{m} \\ \mathbf{m}\end{bmatrix}\right)^\intercal \boldsymbol{\Sigma}_0^{-1}\left(\begin{bmatrix}\boldsymbol{\varphi}_1 \\ \boldsymbol{\varphi}_2\end{bmatrix} - \begin{bmatrix}\mathbf{m} \\ \mathbf{m}\end{bmatrix}\right) \\ &+\frac{1}{2}\log|\boldsymbol{\Sigma}_1| + \frac{1}{2}\left(\begin{bmatrix}\boldsymbol{\varphi}_1 \\ \boldsymbol{\varphi}_2\end{bmatrix} - \begin{bmatrix}\mathbf{m} \\ \mathbf{m}\end{bmatrix}\right)^\intercal \boldsymbol{\Sigma}_1^{-1}\left(\begin{bmatrix}\boldsymbol{\varphi}_1 \\ \boldsymbol{\varphi}_2\end{bmatrix} - \begin{bmatrix}\mathbf{m} \\ \mathbf{m}\end{bmatrix}\right), \end{aligned} \tag{2.43}$$

where the inverse matrices can be expressed explicitly (see Appendix B.2 for derivation):

$$\begin{aligned}\boldsymbol{\Sigma}_0^{-1} &= \frac{1}{2}\begin{bmatrix} \boldsymbol{\Sigma}_{wc}^{-1} + (2\boldsymbol{\Sigma}_{ac} + \boldsymbol{\Sigma}_{wc})^{-1} & -\boldsymbol{\Sigma}_{wc}^{-1} + (2\boldsymbol{\Sigma}_{ac} + \boldsymbol{\Sigma}_{wc})^{-1} \\ -\boldsymbol{\Sigma}_{wc}^{-1} + (2\boldsymbol{\Sigma}_{ac} + \boldsymbol{\Sigma}_{wc})^{-1} & \boldsymbol{\Sigma}_{wc}^{-1} + (2\boldsymbol{\Sigma}_{ac} + \boldsymbol{\Sigma}_{wc})^{-1} \end{bmatrix}, \\ \boldsymbol{\Sigma}_1^{-1} &= \begin{bmatrix} \boldsymbol{\Sigma}_{tot}^{-1} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_{tot}^{-1} \end{bmatrix}.\end{aligned} \tag{2.44}$$

By expanding and simplifying the expression for score computation, we eventually obtain (see Appendix B.3 for details):

$$\boxed{s(\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2) = \boldsymbol{\varphi}_1^\intercal \boldsymbol{\Lambda} \boldsymbol{\varphi}_2 + \boldsymbol{\varphi}_2^\intercal \boldsymbol{\Lambda} \boldsymbol{\varphi}_1 + \boldsymbol{\varphi}_1^\intercal \boldsymbol{\Gamma} \boldsymbol{\varphi}_1 + \boldsymbol{\varphi}_2^\intercal \boldsymbol{\Gamma} \boldsymbol{\varphi}_2 + (\boldsymbol{\varphi}_1 + \boldsymbol{\varphi}_2)^\intercal \mathbf{c} + k,} \tag{2.45}$$

where

$$\boldsymbol{\Lambda} = \frac{1}{4}\boldsymbol{\Sigma}_{wc}^{-1} - \frac{1}{4}(2\boldsymbol{\Sigma}_{ac} + \boldsymbol{\Sigma}_{wc})^{-1} \tag{2.46}$$

$$\boldsymbol{\Gamma} = -\frac{1}{4}(2\boldsymbol{\Sigma}_{ac} + \boldsymbol{\Sigma}_{wc})^{-1} - \frac{1}{4}\boldsymbol{\Sigma}_{wc}^{-1} + \frac{1}{2}\boldsymbol{\Sigma}_{tot}^{-1} \tag{2.47}$$

$$\mathbf{c} = ((2\boldsymbol{\Sigma}_{ac} + \boldsymbol{\Sigma}_{wc})^{-1} - \boldsymbol{\Sigma}_{tot}^{-1})\mathbf{m} \tag{2.48}$$

$$k = \log|\boldsymbol{\Sigma}_{tot}| - \frac{1}{2}\log|\boldsymbol{\Sigma}_{wc}| - \frac{1}{2}\log|2\boldsymbol{\Sigma}_{ac} + \boldsymbol{\Sigma}_{wc}| - \mathbf{m}^\intercal \mathbf{c}. \tag{2.49}$$

### 2.4.3 Cosine Similarity Scoring

As discussed in Section 2.4.1, widespread objective functions for training modern embedding extractors are angular-based, such as AM or AAM. They measure the similarity between embeddings in terms of angle. Therefore, it has become natural to employ the same notion of measure, expressed by cosine similarity, also in the scoring phase. Given unnormalized embeddings $\boldsymbol{\varphi}_1$ and $\boldsymbol{\varphi}_2$, it is computed as:

$$\boxed{s(\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2) = \frac{\boldsymbol{\varphi}_1^\intercal \boldsymbol{\varphi}_2}{\|\boldsymbol{\varphi}_1\|_2 \|\boldsymbol{\varphi}_2\|_2}.} \tag{2.50}$$

We also note that cosine similarity was used even before the introduction of angular-based loss functions [Variani et al., 2014].

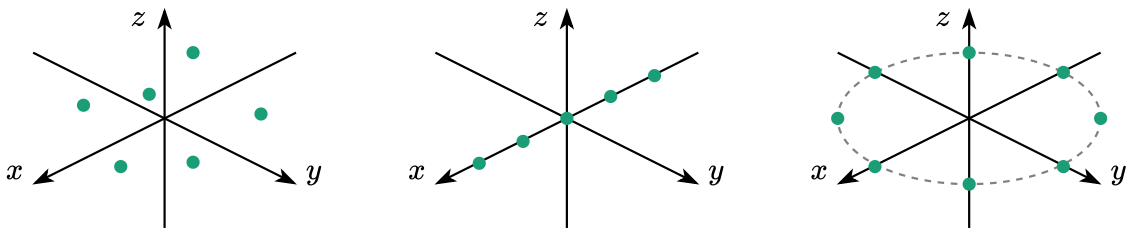
# Chapter 3

# Microphone Array Signal Processing and Spatial Filtering

Multi-channel processing includes various techniques that take multiple signals as input. Signals in speech processing are usually captured by microphones. A set of such simultaneously recording sensors is referred to as a *microphone array*. It can be arranged in a regular topology or ad hoc. Regular arrays are often *planar* or *spherical* (see Figure 3.1 for examples). Microphones in a planar array are located on a 2-dimensional plane. They can form a grid, a triangle, spirals, and other shapes. A special type of microphone array, where sensors lie on a circle and the angle between adjacent microphones is constant, is referred to as a uniform circular array (UCA). In another well-known topology, sensors are equidistantly placed on a line. It is called a uniform linear array (ULA). Generally, the shape of a microphone array changes its properties. The effect of some modifications to the arrangement will be presented later.

Signals from sensors placed at different spatial locations can provide information about acoustic conditions in the recording enclosure and the notion of space in general. It becomes helpful when we want to enhance speech by suppressing noises whose sources occupy spatial locations in the same enclosure. A specific class of multi-channel processing techniques that will lie at the heart of many of our models is *beamforming* — spatial filtering. In general, beamforming attempts to emphasize sounds coming from a direction of interest. At the same time, it suppresses other sources and noise originating from different locations.

We will start our discussion with an introduction to the theoretical background of beamforming, which is based on [Trees, 2002, Benesty et al., 2008, McCowan, 2001, McDonough and Kumatani, 2012]. Then, we will present examples of adaptive beamformers of interest




(a) Ad-hoc array (b) ULA (c) Planar UCA

Figure 3.1: Examples of microphone array geometries.

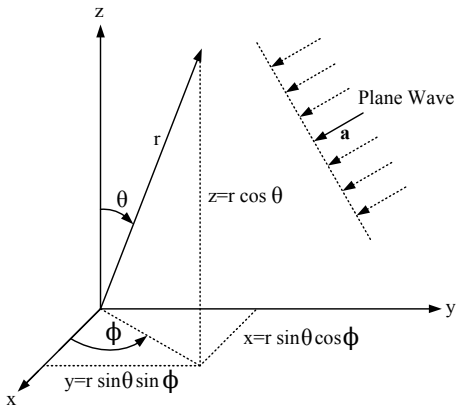



Figure 3.2: Convertibility between the spherical and Cartesian coordinates. The direction of arrival (DoA) of the plane wave is denoted by **a**. Reprinted from [McDonough and Kumatani, 2012].

to us. Finally, we will introduce the concept of broadband beamforming suitable for speech signals.

## 3.1 Beamforming Theory

In principle, microphone arrays are sampled continuous apertures, where the term *aperture* refers to a spatial region that, in our case, receives acoustic waves. In the initial part of this section, we will focus on characterizing properties of continuous apertures.

The discussion will rely on a geometrical representation. Thus, we will first introduce a geometrical model. A point is described by $(x, y, z)$ in the Cartesian coordinate system and by $(r, \theta, \phi)$ in the spherical coordinate system. Both representations are displayed in Figure 3.2. The *polar angle* $\theta$ ranges from 0 to $2\pi$, *azimuth* $\phi \in [-\pi, \pi]$. Coordinate systems are mutually convertible: Cartesian coordinates of vector $\mathbf{r}$ are represented in terms of length $r$, polar angle $\theta$, and azimuth $\phi$ as

$$\mathbf{r} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} r\sin\theta\cos\phi \\ r\sin\theta\sin\phi \\ r\cos\theta \end{bmatrix}. \tag{3.1}$$

Figure 3.2 also displays a plane wave, where the *direction of arrival* (DoA) is indicated by vector[1] $\mathbf{a} = -[\sin\theta\cos\phi, \sin\theta\sin\phi, \cos\theta]^\intercal$. A plane wave approximation is used as it is assumed that the source is distant from the array compared to the size of the array. We aim to describe the propagation of a sound wave under such an approximation. Generally, a differential equation governing wave propagation (not limited to planar sound waves) is called *wave equation*. A solution of the differential wave equation under the plane wave assumption describes the amplitude of a monochromatic wave at a given time and space. It is given as [Ziomek, 1994]:

$$x(t, \mathbf{r}) = Ae^{j(\omega t - \mathbf{k}^\intercal \mathbf{r})}, \tag{3.2}$$

[1]Please note that we use the angle symbols $\theta$ and $\phi$ generally. Therefore, the angles used in (3.1) are not equal to those in $\mathbf{a}$.

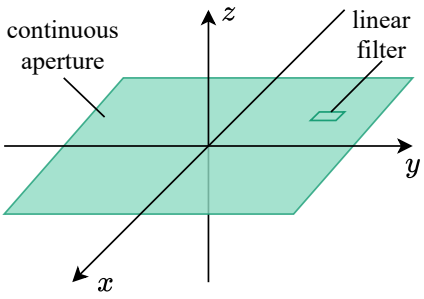



Figure 3.3: Planar continuous aperture is displayed as a green rectangle. Every infinitesimal volume represents a linear filter.

where $A$ is an amplitude, $t$ represents a continuous time, $\omega = 2\pi f$ is an angular frequency of wave, $j := \sqrt{-1}$ denotes a complex unit, and

$$\mathbf{k} := \frac{2\pi}{\lambda}\mathbf{a} \tag{3.3}$$

is a *vector wavenumber*. The wavelength of the propagating wave is denoted by $\lambda$. As follows from (3.3) and Figure 3.2, the vector wavenumber encompasses both DoA (via $\mathbf{a}$) and angular spatial frequency (i.e., angular wavenumber) of the wave. A *scalar wavenumber* is then $||\mathbf{k}|| = \frac{2\pi}{\lambda} = \frac{\omega}{c}$, where $c$ is the speed of sound.

### 3.1.1 Continuous Apertures

A *continuous aperture* is a general term standing for a spatial region that, in our case, receives propagating waves (see Figure 3.3). Having equipped ourselves with the above-mentioned concepts, we will aim to describe an aperture response to propagating waves. Our analysis will then naturally lead to the description of a microphone array response, where we will observe an analogy to the relation between continuous and discrete time-domain signals.

Our goal is to infer an aperture response to a unit *monochromatic* plane wave

$$x(t, \mathbf{r}) = e^{j(\omega t - \mathbf{k}^\intercal \mathbf{r})}. \tag{3.4}$$

Every infinitesimal volume of the aperture may be viewed as a position-specific linear filter characterized by an impulse response $h(t, \mathbf{r})$ where we emphasize dependence on a position by $\mathbf{r}$. A received signal at the position $\mathbf{r}$ is thus given by a convolutional integral,

$$y(t, \mathbf{r}) = \int_{-\infty}^{\infty} x(\tau, \mathbf{r}) h(t - \tau, \mathbf{r}) d\tau = \int_{-\infty}^{\infty} x(t - \tau, \mathbf{r}) h(\tau, \mathbf{r}) d\tau. \tag{3.5}$$

By substituting (3.4) into (3.5), we obtain

$$y(t, \mathbf{r}) = e^{j\omega t} e^{-j\mathbf{k}^\intercal \mathbf{r}} \underbrace{\int_{-\infty}^{\infty} e^{-j\omega\tau} h(\tau, \mathbf{r})\, d\tau}_{\mathcal{F}_\tau\{h(\tau,\mathbf{r})\} = H(\omega,\mathbf{r})}, \tag{3.6}$$

where we identified the Fourier transform of the impulse response $h(\tau, \mathbf{r})$, giving the frequency response $H(\omega, \mathbf{r})$. It is sometimes referred to as *aperture function* or *sensitivity function* [McCowan, 2001]. Overall, we can write that $y(t, \mathbf{r}) = x(t, \mathbf{r})H(\omega, \mathbf{r})$. Assuming the spatial region of the aperture is defined by $\mathcal{R}$, the entire aperture then receives a combination of all individual contributions of infinitesimal volumes:

$$y(t) = \int_{\mathcal{R}} x(t, \mathbf{r})H(\omega, \mathbf{r})d\mathbf{r} = e^{j\omega t} \underbrace{\int_{\mathcal{R}} e^{-j\mathbf{k}^\intercal \mathbf{r}} H(\omega, \mathbf{r})d\mathbf{r}}_{\mathcal{F}_{\mathbf{r}}\{H(\omega,\mathbf{r})\}}. \tag{3.7}$$

Moving the term $e^{j\omega t}$ outside of the integral, we obtained the three-dimensional Fourier transform of the aperture function viewed as a function of position $\mathbf{r}$:

$$\Upsilon(\omega, \mathbf{k}) := \int_{\mathcal{R}} H(\omega, \mathbf{r})e^{-j\mathbf{k}^\intercal \mathbf{r}} d\mathbf{r}, \tag{3.8}$$

which is also termed *frequency-wavenumber response function* [Trees, 2002]. The name emphasizes dependence on the wave angular frequency $\omega$ and the direction coupled with wavelength represented by $\mathbf{k}$. The term $e^{-j\mathbf{k}^\intercal \mathbf{r}}$ is the *array manifold function*, representing a phase shift generated by propagation delay, since we can write that

$$\exp(-j\mathbf{k}^\intercal \mathbf{r}) = \exp\left(-j2\pi f \frac{\mathbf{a}^\intercal}{c} \begin{bmatrix} x \\ y \\ z \end{bmatrix}\right). \tag{3.9}$$

We recall that values of $\mathbf{r}$ are points on the aperture. The frequency-wavenumber response function evaluated versus the direction of arrival is called *beampattern* [Trees, 2002]:

$$\boxed{B(\omega : \theta, \phi) = \Upsilon(\omega, \mathbf{k})\big|_{\mathbf{k}=\frac{2\pi}{\lambda}\mathbf{a}(\theta,\phi)},} \tag{3.10}$$

where $\mathbf{a}(\theta, \phi)$ represents the vector $\mathbf{a}$ evaluated at spherical coordinate angles $\theta$ and $\phi$.

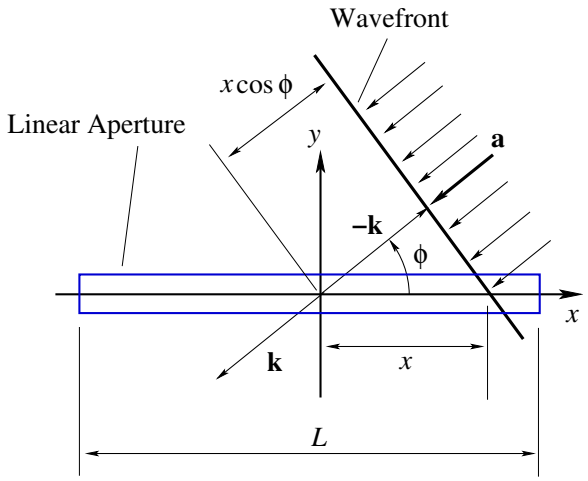



Figure 3.4: Plane wave impinging on a continuous linear aperture (marked by a blue rectangle) lying along the $x$-axis. Reprinted from [McDonough and Kumatani, 2012] and modified.

In order to gain an intuition about the beampattern, we will, without loss of generality, limit our further description to a linear aperture coinciding with the $x$-axis centered around

the origin as displayed in Figure 3.4, resulting in

$$\mathbf{r} = \begin{bmatrix} x \\ 0 \\ 0 \end{bmatrix}. \tag{3.11}$$

The ideas presented are extendable to the 3-dimensional case. In the simplified 1-dimensional case, it is easier to see how the phase shift depends on the wavenumber and position outlined in (3.9). The argument $\mathbf{k}^\intercal\mathbf{r}$ simplifies as

$$\mathbf{k}^\intercal\mathbf{r} = k_x x = 2\pi f \frac{-\sin\theta\cos\phi}{c} x. \tag{3.12}$$

Considering that each point at the aperture receives a delayed signal $x(t-\tau_x)$ and following the geometrical representation in Figure 3.4, the time delay can be expressed as

$$\tau_x = -\frac{x\cos\phi}{c}. \tag{3.13}$$

Realizing that the polar angle $\theta = \pi/2$ and substituting (3.13) into (3.12), we get

$$k_x x = \omega\tau_x. \tag{3.14}$$

Assuming that the aperture is centered around the origin and it has a length of $L$, the beampattern simplifies to

$$B(\omega:\theta,\phi) = \int_{-L/2}^{L/2} H(\omega,x)e^{-jk_x x}dx. \tag{3.15}$$

Now, let us consider that the aperture function is uniform and does not depend on angular frequency $H(\omega,x) = \frac{1}{L}$. The beampattern is then

$$B(\omega:\theta,\phi) = \frac{1}{L}\int_{-L/2}^{L/2} e^{-jk_x x}dx, \tag{3.16}$$

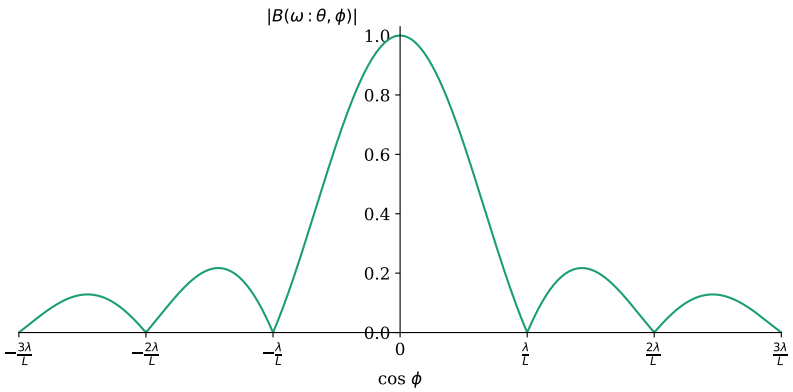



Figure 3.5: Beampattern of a linear aperture of length $L$ lying on the x-axis, centered around the origin. Polar angle $\theta = \frac{\pi}{2}$.

which has a well-known solution

$$B(\omega : \theta, \phi) = \text{sinc}\left(\frac{L}{2}k_x\right) = \text{sinc}\left(\frac{\pi L}{\lambda}\sin\theta\cos\phi\right). \tag{3.17}$$

The beampattern is displayed in Figure 3.5 as a function of $\cos\phi$ for $\theta = \pi/2$. The part of it ranging from $-\lambda/L$ to $\lambda/L$ is called *main lobe* [McCowan, 2001], and its range is referred to as *beam width*. Lobes other than the main one are *side lobes*. From Figure 3.5, it is clear that the aperture is the most sensitive to waves impinging at the azimuth of $\phi = \frac{\pi}{2}$ (*broadside* sources). Rewriting the beam width as $2\frac{c}{fL}$, we can conclude that the main lobe extent is inversely proportional to both the aperture length and the temporal frequency of the propagating wave.

### 3.1.2 Microphone Arrays

Discrete signals are obtained by sampling continuous signals. Drawing an analogy, microphone arrays are, in essence, spatially sampled continuous apertures. In order to analyze the beam pattern of microphone arrays, we will build upon the previous discussion. Therefore, the ULA will be the array geometry of interest. In ULAs, microphones are distributed along the $x$-axis, and they are assumed to be ideal and omnidirectional. Let us consider that an odd number $N$ of sensors uniformly sample the aperture (as shown in Figure 3.6) where the inter-sensor distance is $d$. An array weighting function is then obtained from the aperture function by sampling:

$$\sum_{n=-(N-1)/2}^{(N-1)/2} \delta(x - nd)H(\omega, x). \tag{3.18}$$

Plugging the array weighting function into the aperture beam pattern formula in (3.15), we obtain

$$\begin{aligned} B_{\text{arr}}(\omega : \theta, \phi) &= \int_{-L/2}^{L/2} \sum_n \left[\delta(x - nd)H(\omega, x)\right] e^{-jk_x x} dx \\ &= \sum_n H(\omega, nd) e^{-jk_x nd}. \end{aligned} \tag{3.19}$$

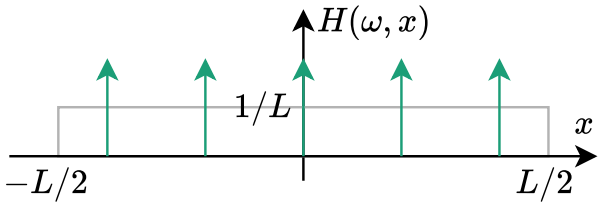
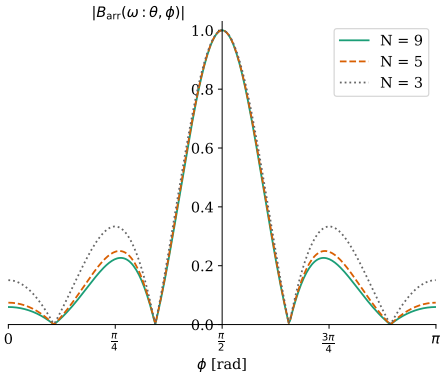
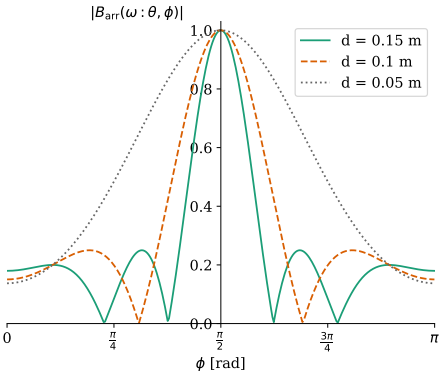



Figure 3.6: Sampling of a continuous linear aperture with a uniform response with Dirac impulses.




(a) Different number of sensors keeping $Nd = 0.6\,\text{m}$, $f = 1.2\,\text{kHz}$.




(b) Different inter-sensor distance, $f = 1.2\,\text{kHz}$, $N = 5$.

Figure 3.7: Impact of microphone array properties on the beampattern.

Expressing $k_x$ in terms of the (angular) temporal[2] frequency and defining $w_n(\omega) := H(\omega, nd)$, we can rewrite the beampattern as

$$B_{\text{arr}}(\omega:\theta,\phi) = \sum_n w_n(\omega) e^{j\omega \frac{nd}{c} \sin\theta \cos\phi}. \tag{3.20}$$

The beampattern depends on a few variables, such as the number of elements in the array $N$, the spacing between elements $d$, and the angular frequency of an impinging wave $\omega$ (or $f$). Figure 3.7a shows the effect of the sensor count while keeping the *effective length*[3] $Nd$ constant. The increase in samples (microphones) reduces the height of sidelobes, making sounds coming from non-broadside locations less pronounced. The impact of inter-microphone spacing, while keeping the number of elements $N$ constant, is presented in Figure 3.7b. A beam width increases with decreasing distance between sensors.

In speech processing, the frequency dependence of the beampattern is of great importance since speech is a wideband signal. Figure 3.8 demonstrates that the beam width decreases with increasing frequency. The result of this behavior is that a distracting source coming from a direction different from the look direction is attenuated non-uniformly across frequencies. Microphone array processing for wideband speech signals will be further discussed in Section 3.3.

The beampattern formula for a microphone array sampling a linear aperture on the x-axis in (3.19) can be generalized to the case of sampling volume apertures[4]:

$$B_{\text{arr}}(\omega:\theta,\phi) = \sum_n H(\omega, \mathbf{r}_n) e^{-j\mathbf{k}^\intercal \mathbf{r}_n} = \sum_n w_n(\omega) e^{-j\mathbf{k}^\intercal \mathbf{r}_n}. \tag{3.21}$$

Analogously to continuous apertures, the term $e^{-j\mathbf{k}^\intercal \mathbf{r}_n}$, incorporating spatial characteristics of the array, is called *array manifold vector*. The formula can be used to analyze microphone arrays of various geometries. Assuming a uniform weighting $H(\omega, \mathbf{r}_n) = \frac{1}{N}$, we demonstrate the impact of different array geometries (examples of plane sampling) on their response in

[2] "Temporal" to distinguish it from the spatial frequency.

[3] The effective length corresponds to a span of the continuous aperture, which is sampled.

[4] Not to clutter notation, here we assume $w_n(\omega) := H(\omega, \mathbf{r}_n)$, which is a mere extension of $w_n(\omega)$ in (3.20) to arbitrarily located microphones in a 3-dimensional space.

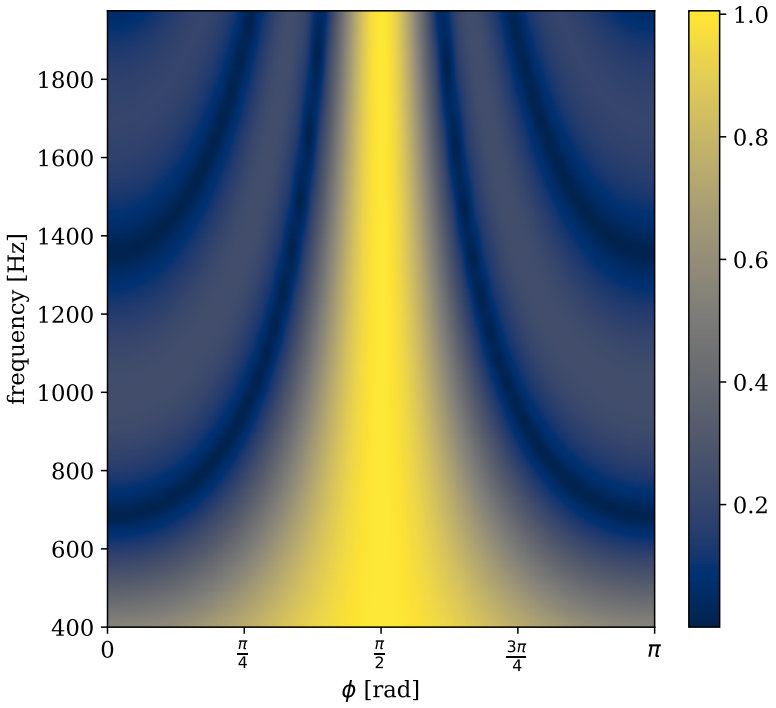


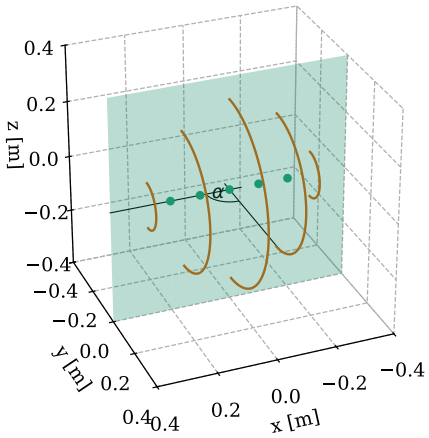

Figure 3.8: Impact of a temporal frequency $f$ on the beampattern. It ranges from 400 to 2000 Hz. A uniform linear array with $N = 5$ elements and inter-sensor distance $d = 0.1$ m was used.

Figure 3.9. To better understand the arrays' sensitivity, we do not directly vary $\phi$ and $\theta$ angles. Instead, every column in the response image corresponds to an orange half-circle, representing part of a spherical circle with a constant rotation angle $\alpha$, through which waves impinge on the array. For example, for a linear array (Figure 3.9a), we can observe insensitivity to rotation around the x-axis, which is tackled by 2D planar (Figure 3.9c), and circular (Figure 3.9e) arrays.

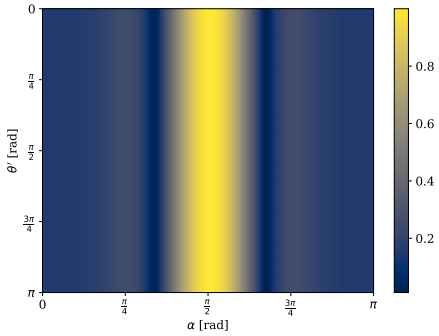
### 3.1.3 Beamforming

In Section 3.1.2, we saw that modifications of the geometry of a microphone array shape the beampattern. However, in practice, a microphone array geometry is often fixed and cannot be changed while deployed. Nevertheless, we still want to be able to modify the beampattern. So far, we assumed that all microphones were equal, having $w_n(\omega) = \frac{1}{N}$. By applying weights to microphones, we change their output characteristics. It can be equally viewed as a change to $w_n(\omega)$, referred to as *beamforming*.

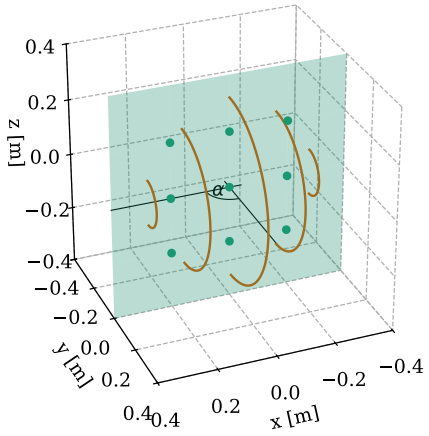
In the first part of this section, we will build intuition behind beamforming. To this end, we will limit our discussion to a linear array lying on the x-axis, noting that our findings are generalizable to arrays of arbitrary geometry. In Figure 3.7, the maxima of beampatterns were centered around azimuth $\phi = \frac{\pi}{2}$, meaning that the array is the most sensitive to waves arriving from broadside. This behavior is unfavorable when the source of interest position is characterized by another direction, for instance, $\phi' = \frac{\pi}{3}$ as shown in Figure 3.10. In such a case, we want to shift the beampattern to the left, so that the maximum aligns with $\phi = \frac{\pi}{3}$. Since a goniometric function of cos monotonically decreases in the interval from 0



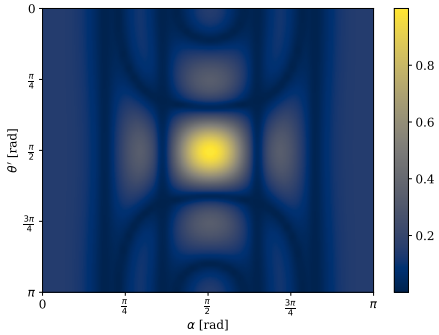

(a) Linear array geometry.



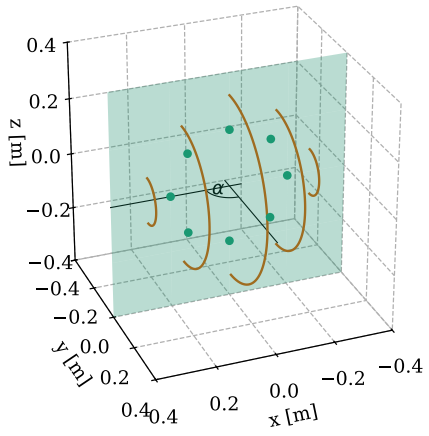

(b) Linear array response.



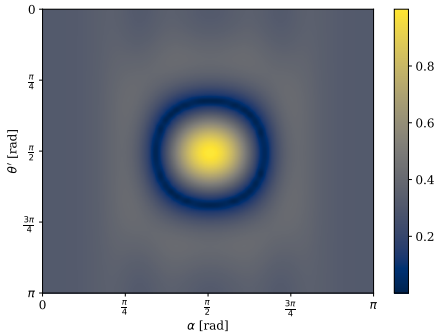

(c) Planar array geometry.




(d) Planar array response.




(e) Circular array geometry.




(f) Circular array response.

Figure 3.9: Different array geometries and corresponding responses to waves with a temporal frequency of 1.2 kHz. In each response plot, the horizontal angle of arrival $\alpha$ and vertical angle of arrival $\theta$ are unwrapped on the horizontal and vertical axes, respectively, and the orange circles denote how we measure $\theta$.

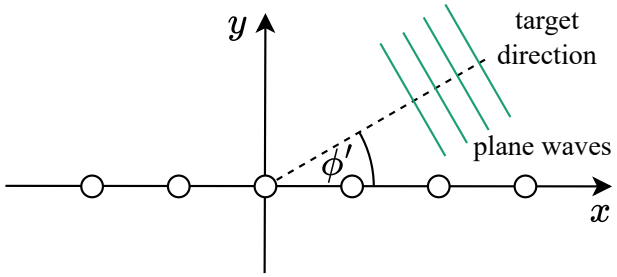



Figure 3.10: Plane wave impinging on the microphone array at an (azimuth) angle of $\phi' = \frac{\pi}{3}$.

to $\pi$, we can reach the required properties of the beampattern by modifying (3.20), treating it as a function of $\cos\phi$, as follows:

$$B'_{\text{arr}}(\omega : \theta, \phi) = \sum_n \frac{1}{N} e^{j\omega \frac{nd}{c}(\cos\phi - cos\phi')}. \tag{3.22}$$

Using result in (3.13), we can further rewrite it as

$$\begin{aligned} B'_{\text{arr}}(\omega : \theta, \phi) &= \sum_n \frac{1}{N} e^{-j\omega \frac{nd}{c} \cos\phi'} e^{j\omega \frac{nd}{c} \cos\phi} \\ &= \sum_n \underbrace{\frac{1}{N} e^{j\omega\tau'_n}} e^{j\omega \frac{nd}{c} \cos\phi}, \end{aligned} \tag{3.23}$$

where we can identify that $w_n(\omega) = \frac{1}{N} e^{j\omega\tau'_n}$ is a complex weight of the $n$th microphone. The described change of the beampattern induced by complex weighting is visualized in Figure 3.11. We conclude that by modifying $\angle w_n(\omega)$, we change the angular location of the main lobe, which is called *steering*. The absolute value $|w_n(\omega)|$ gives importance to microphones. Different $|w_n(\omega)|$ for different microphones results in a change of the beampattern shape, hence the term *shaping* [McCowan, 2001]. Viewed in the frequency domain, the term $e^{j\omega\tau'_n}$ represents a phase shift that translates to a delay/advance in the time domain. In our particular case, a negative $\tau'_n$ corresponds to a positive position $x$, resulting in delaying signals that were captured by the microphones closer to the source. At the same time, signals captured by the farthest microphone with respect to the wavefront will be advanced. Denoting a spectrum of a signal captured by the $n$th microphone as $Y_n(\omega)$, the output of the described beamforming is

$$\hat{X}(\omega) = \frac{1}{N} \sum_n Y_n(\omega) e^{j\omega\tau'_n}. \tag{3.24}$$

Our discussion revealed the simplest beamforming technique, *delay-and-sum*, which emphasizes the source at the look direction by constructive interference and suppresses other directions via destructive interference.

This section presented how a simple delay-and-sum beamformer arose, given our requirements. In the following text, we will describe more elaborate beamforming techniques.

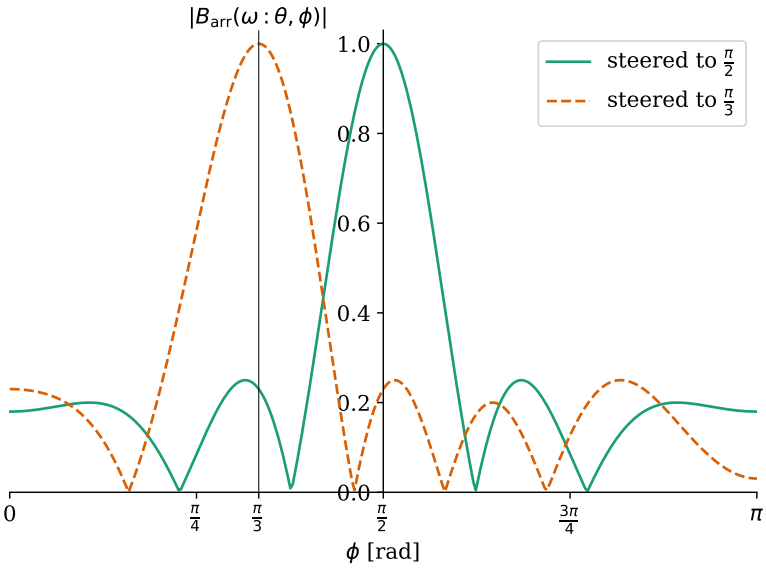



Figure 3.11: Steering a look direction of the beampattern to $\frac{\pi}{3}$ achieved by applying complex weights to microphones. $N = 5$, $d = 0.15\,\text{m}$, and $f = 1.2\,\text{kHz}$.

## 3.2 Adaptive Beamforming

Section 3.1.3 introduced the core idea of beamforming and the impact of assigning complex weights to sensors. As a result of our analysis, we derived a delay-and-sum beamformer. In order to compute beamforming weights, only a delay at the microphones was required, making the design deterministic. Therefore, delay-and-sum is a representative technique of a class of *data-independent* beamformers (also referred to as *fixed* beamformers). Limitations of the fixed beamformers are tackled by solutions that derive information about sources from the observed data, called *data-dependent* (or *adaptive*) beamformers [McCowan et al., 2001,Benesty et al., 2008]. They represent a potentially more powerful alternative, provided that source-related estimates are reliable and reflect reality. As a result, they are more practical and often widely adopted.

In this section, we will describe objectives and derivation of two major beamformers, Minimum Variance Distortionless Response (MVDR) filter [Capon, 1969] and Generalized Eigenvalue (GEV) beamformer [Warsitz and Haeb-Umbach, 2007]. Since they are defined by optimizing specific criteria given input data, it is worthwhile to start with a description of a signal model.

### 3.2.1 Signal Model

In Section 3.1, we assumed continuous signals. We will now concentrate on their sampled versions, discrete signals. Let us assume that the microphone array comprises $N$ sensors and frequency-domain signals, obtained by the DTFT, are modeled as

$$\begin{aligned} Y_n(\omega) &= X_n(\omega) + V_n(\omega) \\ &= S(\omega)G_n(\omega) + V_n(\omega), n = 1, 2, \ldots, N, \end{aligned} \tag{3.25}$$

where $X_n(\omega)$, $V_n(\omega)$, and $Y_n(\omega)$ denote spectral images of reverberant speech, additive noise, and the signal captured at the $n$th microphone, respectively. We will consider the reverberant speech at the reference microphone as our target. $S(\omega)$ represents the source-signal spectrum. A channel between the source and the $n$th microphone is represented by a linear filter, where $G_n(\omega)$ is a DTFT image of an impulse response or relative transfer function (RTF). It represents the reverberation effects. In the following text, it will be purposeful to use a vector notation. Therefore, the same model can be equivalently expressed as

$$\begin{aligned} \mathbf{y}(\omega) &= \mathbf{x}(\omega) + \mathbf{v}(\omega), \\ \mathbf{y}(\omega) &= S(\omega)\mathbf{g}(\omega) + \mathbf{v}(\omega), \end{aligned} \tag{3.26}$$

where we define

$$\begin{aligned} \mathbf{y}(\omega) &= [Y_1(\omega), Y_2(\omega), \ldots, Y_N(\omega)]^\mathsf{T}, \\ \mathbf{x}(\omega) &= S(\omega)[G_1(\omega), G_2(\omega), \ldots, G_N(\omega)]^\mathsf{T} \\ &= S(\omega)\mathbf{g}(\omega). \end{aligned} \tag{3.27}$$

The noise vector $\mathbf{v}(\omega)$ is defined analogically to $\mathbf{y}(\omega)$. In accordance with our result in (3.24), beamforming, which provides an estimate of the speech image, can be expressed as a *linear combination*:

$$\hat{X}(\omega) = \mathbf{w}^\mathsf{H}(\omega)\mathbf{y}(\omega), \tag{3.28}$$

where $\mathbf{w}^\mathsf{H}(\omega)$ denotes an $N$-dimensional vector of frequency-dependent beamforming weights. The following sections will be dedicated to approaches to obtaining optimal beamforming weights in a statistical sense. We note that the description will consider one specific angular frequency $\omega$.

### 3.2.2 Minimum Variance Distortionless Response

Minimum Variance Distortionless Response (MVDR) beamformer [Capon, 1969] is one of the most widely adopted multi-channel spatial filters. In this section, we will first aim to derive the traditional MVDR beamforming weights, where a non-reverberant environment is expected. Then, we will turn to the formulation that considers reverberation and thus conforms to our signal model. Finally, we will modify the result to a more practical one, which is also used in our experiments.

In the ideal conditions of a *non-reverberant* environment, the signal impinging on the array can be given as $S(\omega)\mathbf{d}(\omega, \mathbf{k})$ with $\mathbf{d}(\omega, \mathbf{k})$ denoting the array manifold vector introduced in (3.21). We consider it the target signal in this particular case. Note the difference compared to the reverberant speech component (3.26). In the presence of noise, the output of the beamformer according to (3.28) is then

$$\begin{aligned} \hat{X}(\omega) &= \mathbf{w}^\mathsf{H}(\omega)[S(\omega)\mathbf{d}(\omega, \mathbf{k}) + \mathbf{v}(\omega)] \\ \hat{X}(\omega) &= \mathbf{w}^\mathsf{H}(\omega)S(\omega)\mathbf{d}(\omega, \mathbf{k}) + \mathbf{w}^\mathsf{H}(\omega)\mathbf{v}(\omega). \end{aligned} \tag{3.29}$$

As suggested by the name, the key to the MVDR is the distortionless criterion. It requires that any waveform coming from the target direction will pass through the filter without distortion. Mathematically, in the absence of noise, the following equality should hold:

$$\hat{X}(\omega) = S(\omega). \tag{3.30}$$

Considering filtering applied in (3.29), we can rewrite the distortionless requirement in (3.30) as

$$\mathbf{w}^\mathsf{H}(\omega)S(\omega)\mathbf{d}(\omega, \mathbf{k}) = S(\omega), \tag{3.31}$$

which directly leads to the well-known *distortionless constraint*: $\mathbf{w}^{\mathsf{H}}(\omega)\mathbf{d}(\omega,\mathbf{k}) = 1$. The MVDR minimizes the variance (energy) of the output $\hat{X}(\omega)$ in the presence of additive noise under such a constraint. It translates to minimzation of the filtered noise variance: $E[|\mathbf{w}^{\mathsf{H}}(\omega)\mathbf{v}(\omega)|^2]$, where the expectation is over realizations. Finally, the objective to minimize is

$$\mathbf{w}^{\mathsf{H}}(\omega)\mathbf{\Phi}^{(vv)}(\omega)\mathbf{w}(\omega) \quad \text{s.t. } \mathbf{w}^{\mathsf{H}}(\omega)\mathbf{d}(\omega,\mathbf{k}) = 1, \tag{3.32}$$

where $\mathbf{\Phi}^{(vv)}(\omega) = E[\mathbf{v}(\omega)\mathbf{v}^{\mathsf{H}}(\omega)]$ is a covariance matrix (assuming a mean of zero) referred to as *power spectral density (PSD) matrix* [Benesty et al., 2008, Souden et al., 2010, Heymann et al., 2016a], spatial spectral covariance matrix [McDonough and Kumatani, 2012], or simply *spatial covariance matrix (SCM)* [Erdogan et al., 2016, Ochiai et al., 2020]. Using a Lagrange multiplier, we can search for the minimum by

$$\frac{\partial}{\partial \mathbf{w}^{\mathsf{H}}}\left\{\mathbf{w}^{\mathsf{H}}(\omega)\mathbf{\Phi}^{(vv)}(\omega)\mathbf{w}(\omega) + \lambda(\omega)[\mathbf{w}^{\mathsf{H}}(\omega)\mathbf{d}(\omega,\mathbf{k}) - 1] + \lambda^*(\omega)[\mathbf{d}^H(\omega,\mathbf{k})\mathbf{w}(\omega) - 1]\right\} = 0. \tag{3.33}$$

Performing derivations, we obtain

$$\mathbf{\Phi}^{(vv)}(\omega)\mathbf{w}(\omega) + \lambda(\omega)\mathbf{d}(\omega,\mathbf{k}) = 0 \tag{3.34}$$

$$\mathbf{\Phi}^{(vv)}(\omega)\mathbf{w}(\omega) = -\lambda(\omega)\mathbf{d}(\omega,\mathbf{k}) \tag{3.35}$$

$$\mathbf{w}(\omega) = -\lambda(\omega)\mathbf{\Phi}^{(vv)^{-1}}(\omega)\mathbf{d}(\omega,\mathbf{k}). \tag{3.36}$$

In order to evaluate $\lambda(\omega)$, we will multiply the last equation by $\mathbf{d}^H(\omega,\mathbf{k})$ and we will make use of the constraint in (3.32):

$$\mathbf{d}^H(\omega,\mathbf{k})\mathbf{w}(\omega) = -\lambda(\omega)\mathbf{d}^H(\omega,\mathbf{k})\mathbf{\Phi}^{(vv)^{-1}}(\omega)\mathbf{d}(\omega,\mathbf{k}) \tag{3.37}$$

$$1 = -\lambda(\omega)\mathbf{d}^H(\omega,\mathbf{k})\mathbf{\Phi}^{(vv)^{-1}}(\omega)\mathbf{d}(\omega,\mathbf{k}) \tag{3.38}$$

$$\lambda(\omega) = -1/[\mathbf{d}^H(\omega,\mathbf{k})\mathbf{\Phi}^{(vv)^{-1}}(\omega)\mathbf{d}(\omega,\mathbf{k})]. \tag{3.39}$$

Finally, plugging the expression for $\lambda(\omega)$ into (3.36), we obtain the well-known formula for the MVDR weight vector computation:

$$\boxed{\mathbf{w}(\omega) = \frac{\mathbf{\Phi}^{(vv)^{-1}}(\omega)\mathbf{d}(\omega,\mathbf{k})}{\mathbf{d}^H(\omega,\mathbf{k})\mathbf{\Phi}^{(vv)^{-1}}(\omega)\mathbf{d}(\omega,\mathbf{k})}.} \tag{3.40}$$

It is worth emphasizing that the MVDR can be seen as an extension of the delay-and-sum beamformer. If the noise component is neglected by setting $\mathbf{\Phi}^{(vv)}(\omega) = \mathbf{I}$, (3.40) simplifies to $\mathbf{w}(\omega) = \mathbf{d}(\omega,\mathbf{k})/N$. Due to the definition of the array manifold vector, applying such a weight vector to the microphone signals corresponds to scaling and delaying. It matches the definition of the delay-and-sum beamformer derived in Section 3.1.3.

As noted before (and reflected in the equations), the original assumption is that only primary (sound) waves (without reflections) impinge on the array. In practical situations, microphone arrays and the source of the target signal are usually placed in reverberant enclosures (such as rooms). The signal model corresponding to these scenarios was described in Section 3.2.1. In the presence of reverberation, the condition on undistorted speech in (3.30) changes. It is now required that beamforming, in a noise-free environment, recovers

the (reverberant) speech signal image at the reference[5] microphone [Benesty et al., 2008, Souden et al., 2010]:

$$\hat{X}(\omega) = S(\omega)G_1(\omega). \tag{3.41}$$

Having such an assumption specified, we can formulate a distortionless constraint as

$$\begin{aligned} \mathbf{w}^{\mathsf{H}}(\omega)S(\omega)\mathbf{g}(\omega) &= S(\omega)G_1(\omega), \\ \mathbf{w}^{\mathsf{H}}(\omega)\mathbf{g}(\omega) &= G_1(\omega). \end{aligned} \tag{3.42}$$

Under such a constraint, minimization of the noise variance following our previous derivation steps yields the following MVDR formula:

$$\boxed{\mathbf{w}(\omega) = G_1^*(\omega)\frac{{\boldsymbol{\Phi}^{(vv)}}^{-1}(\omega)\mathbf{g}(\omega)}{\mathbf{g}^{\mathsf{H}}(\omega){\boldsymbol{\Phi}^{(vv)}}^{-1}(\omega)\mathbf{g}(\omega)}.} \tag{3.43}$$

In order to obtain the beamforming weights, estimation of the noise second-order statistics is required. They can be predicted during speech inactivity, provided by voice activity detection (VAD). There are also other ways we do not elaborate on in this section, as they will constitute much of the discussion in Section 5.6. The difficulty with the aforementioned formulation of the MVDR beamformer is the requirement of knowing the transfer functions $\mathbf{g}(\omega)$ whose estimation is troublesome in practice [Benesty et al., 2008]. Therefore, we will also show a more practical alternative formulation taking into account second-order statistics of the target signal [Souden et al., 2010]. Analyzing our signal model in (3.26), we can realize that the SCM of the reverberant speech is

$$\boldsymbol{\Phi}^{(xx)}(\omega) = E[S(\omega)\mathbf{g}(\omega)\mathbf{g}^{\mathsf{H}}(\omega)S^*(\omega)] = \phi^{(ss)}(\omega)\mathbf{g}(\omega)\mathbf{g}^{\mathsf{H}}(\omega), \tag{3.44}$$

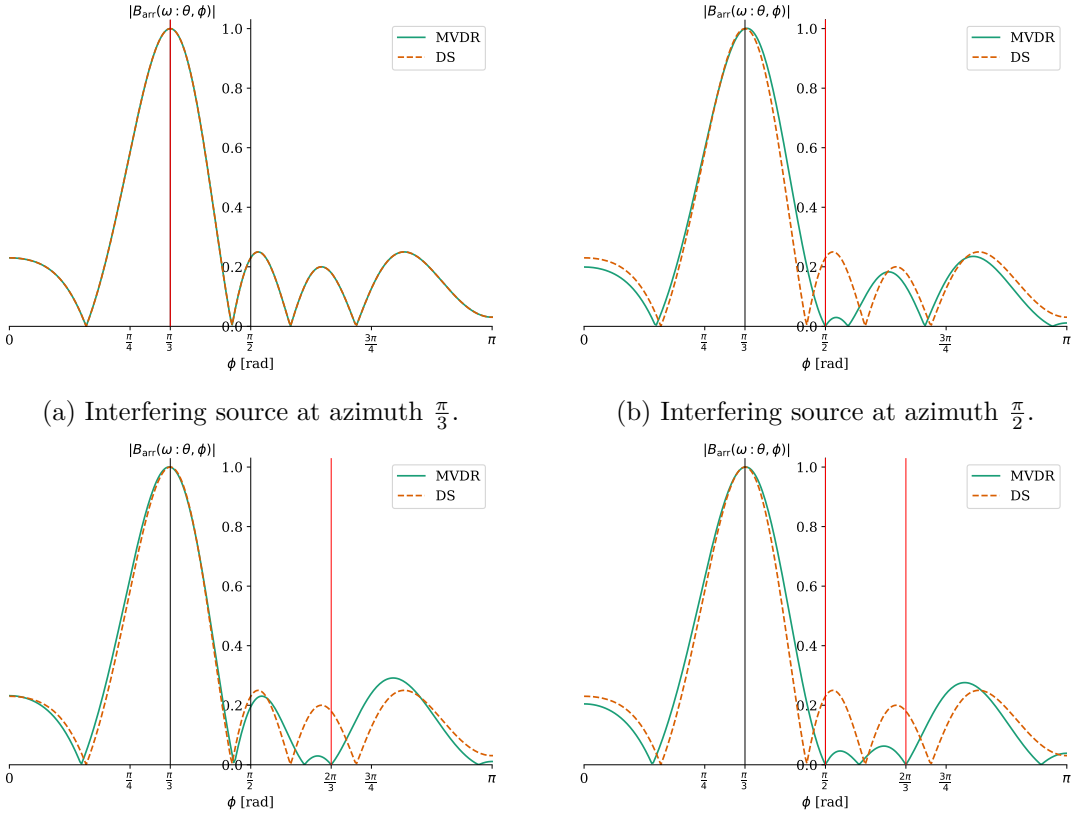
with $\phi^{(ss)}(\omega) = E[S(\omega)S^*(\omega)]$ denoting power spectral density of the source speech signal. To derive the alternative version, we make use of the equality $\mathbf{g}^{\mathsf{H}}(\omega){\boldsymbol{\Phi}^{(vv)}}^{-1}(\omega)\mathbf{g}(\omega) = \mathrm{tr}\{{\boldsymbol{\Phi}^{(vv)}}^{-1}(\omega)\mathbf{g}(\omega)\mathbf{g}^{\mathsf{H}}(\omega)\}$, with $\mathrm{tr}\{\cdot\}$ representing the trace operator. Then, substituting the denominator and multiplying both the numerator and the denominator of (3.43) by $\phi^{(ss)}(\omega)$, we get

$$\begin{aligned} \mathbf{w}(\omega) &= G_1^*(\omega)\frac{{\boldsymbol{\Phi}^{(vv)}}^{-1}(\omega)\mathbf{g}(\omega)\phi^{(ss)}(\omega)}{\mathrm{tr}\{{\boldsymbol{\Phi}^{(vv)}}^{-1}(\omega)\mathbf{g}(\omega)\mathbf{g}^{\mathsf{H}}(\omega)\}\phi^{(ss)}(\omega)} \\ &= G_1^*(\omega)\frac{{\boldsymbol{\Phi}^{(vv)}}^{-1}(\omega)\mathbf{g}(\omega)\phi^{(ss)}(\omega)}{\mathrm{tr}\{{\boldsymbol{\Phi}^{(vv)}}^{-1}(\omega)\boldsymbol{\Phi}^{(xx)}(\omega){\phi^{(ss)}}^{-1}(\omega)\}\phi^{(ss)}(\omega)}, \end{aligned} \tag{3.45}$$

where we employed the expression for the reverberant speech signal SCM, i.e., (3.44), in the denominator. Realizing that $\phi^{(ss)}(\omega)\mathbf{g}(\omega)G_1^*(\omega)$ is the first column of $\boldsymbol{\Phi}^{(xx)}(\omega)$, because the reference microphone has an index of 1 in our analysis, we can rewrite the beamforming weight vector to the final form:

$$\boxed{\mathbf{w}(\omega) = \frac{{\boldsymbol{\Phi}^{(vv)}}^{-1}(\omega)\boldsymbol{\Phi}^{(xx)}(\omega)}{\mathrm{tr}\{{\boldsymbol{\Phi}^{(vv)}}^{-1}(\omega)\boldsymbol{\Phi}^{(xx)}(\omega)\}}\mathbf{u}.} \tag{3.46}$$

[5] Without loss of generality and to ease notation, we assume that the reference microphone has an index of 1.




(a) Interfering source at azimuth $\frac{\pi}{3}$.

(b) Interfering source at azimuth $\frac{\pi}{2}$.

(c) Interfering source at azimuth $\frac{2\pi}{3}$.

(d) Interfering sources at azimuth $\frac{\pi}{2}$ and $\frac{2\pi}{3}$.

Figure 3.12: Comparison of beampatterns of the delay-and-sum (DS) beamformer and the MVDR. Both beamformers are steered to an azimuth of $\frac{\pi}{3}$, and DS equals that in Figure 3.11. The ULA array comprises five microphones with an inter-sensor distance of 15 cm; the frequency was set to 1200 Hz. We simulated various scenarios with interfering source waves arriving from different angles. We also demonstrate the capability of dealing with more than one interfering source.

The vector $\mathbf{u} \in \{0, 1\}^N$ is a one-hot vector encoding the reference microphone. The derived formula is more practical than (3.43) [Benesty et al., 2008]. Having obtained an estimate of $\mathbf{\Phi}^{(vv)}(\omega)$ using VAD and assuming that the target signal and noise are uncorrelated, a straightforward approach to obtaining an estimate of the target signal SCM is by setting $\mathbf{\Phi}^{(xx)}(\omega) = \mathbf{\Phi}^{(yy)}(\omega) - \mathbf{\Phi}^{(vv)}(\omega)$. Some studies also use an estimate of $\mathbf{\Phi}^{(xx)}(\omega)$ to obtain an approximation of $\mathbf{g}(\omega)$ required in the MVDR formula in (3.43) [Taherian et al., 2020]. In such a case, the property of the target signal SCM resulting from (3.44) is utilized: Theoretically, $\mathbf{\Phi}^{(xx)}(\omega)$ should have only one non-zero eigenvalue (as its rank is 1) and the corresponding eigenvector is $\mathbf{g}(\omega)$. Since the target signal SCM prediction might be imprecise in practice, the obtained estimate of $\mathbf{g}(\omega)$ ends up being erroneous as well. With the aim of improving estimates of $\mathbf{\Phi}^{(xx)}(\omega)$, [Serizel et al., 2014] proposed several techniques to obtain a rank-1 approximation of the target signal SCM. As it follows from our discussion, a precise estimation of the second-order statistics is a key to obtaining reliable beamforming weights. Therefore, in this work, we devote much space to describing ways of predicting them in Section 5.6.

In order to demonstrate the noise cancellation ability of the MVDR beamformer, we show beampatterns for scenarios with interfering source waves arriving from various angles in Figure 3.12. We compare the MVDR with delay-and-sum (DS), where both are steered toward an azimuth of $\frac{\pi}{3}$. To create beampatterns, we assumed a reverberation-free environment and followed (3.40). The array manifold vector $\mathbf{d}(\omega, \mathbf{k})$ in the formula is an unnormalized version of the DS beamforming weights. We set noise SCM to $\mathbf{\Phi}^{(vv)}(\omega) := \mathbf{d}_v(\omega)\mathbf{d}_v^{\mathsf{H}}(\omega)$ with $\mathbf{d}_v(\omega)$ representing an alternative to the array manifold vector for noise (encoding delays of noise for individual microphones). Since the rank of such a matrix is one, the required inverse matrix does not exist. In order to obtain a pseudoinverse matrix, we performed a diagonal loading [Woelfel and McDonough, 2009], which adds small values to the diagonal of the SCM to break the linear dependence of rows. Such a technique is also used in practice to avoid numerical instability. The noise SCM for the case with two interfering sources (Figure 3.12d) is obtained as the sum of the SCMs of individual distractors. Due to the distortionless constraint, the beampattern remains unchanged w.r.t. the DS when the target and interfering sources' azimuths coincide. When source waves impinge on the array from distinct angles, we can observe that the beam pattern changes so that it has a response of 0 for the undesired angles.

### 3.2.3 Generalized Eigenvalue Beamformer

Another adaptive spatial filter we will use is termed the generalized eigenvalue (GEV) beamformer. Like MVDR, GEV is an optimal beamformer in a statistical sense, but results from optimizing a different criterion: signal-to-noise ratio (SNR). Assuming our signal model in Section 3.2.1, the power spectral density of the beamformer output can be written as

$$\mathbf{w}^{\mathsf{H}}(\omega)\mathbf{\Phi}^{(xx)}(\omega)\mathbf{w}(\omega) + \mathbf{w}^{\mathsf{H}}(\omega)\mathbf{\Phi}^{(vv)}(\omega)\mathbf{w}(\omega). \tag{3.47}$$

Then, the SNR criterion that we aim to optimize is formulated as

$$\mathrm{SNR}(\omega) = \frac{\mathbf{w}^{\mathsf{H}}(\omega)\mathbf{\Phi}^{(xx)}(\omega)\mathbf{w}(\omega)}{\mathbf{w}^{\mathsf{H}}(\omega)\mathbf{\Phi}^{(vv)}(\omega)\mathbf{w}(\omega)} \tag{3.48}$$

and hence

$$\mathbf{w}(\omega) = \arg\max_{\mathbf{w}(\omega)} \frac{\mathbf{w}^{\mathsf{H}}(\omega)\mathbf{\Phi}^{(xx)}(\omega)\mathbf{w}(\omega)}{\mathbf{w}^{\mathsf{H}}(\omega)\mathbf{\Phi}^{(vv)}(\omega)\mathbf{w}(\omega)}. \tag{3.49}$$

Since both $\mathbf{\Phi}^{(xx)}(\omega)$ and $\mathbf{\Phi}^{(vv)}(\omega)$ are Hermitian matrices and assuming that the noise SCM is non-singular, the expression in (3.48) is, in fact, a generalized Rayleigh quotient. Therefore, its values $\lambda(\omega)$ (corresponding to the eigenvalues) satisfy

$$\boxed{\mathbf{\Phi}^{(xx)}(\omega)\mathbf{w}(\omega) = \lambda(\omega)\mathbf{\Phi}^{(vv)}(\omega)\mathbf{w}(\omega),} \tag{3.50}$$

which is called the generalized eigenvalue problem, hence the beamformer name. Then the maximum $\lambda(\omega)$ of the generalized Rayleigh quotient (and SNR) corresponds to the largest eigenvalue of $\mathbf{\Phi}^{(vv)^{-1}}(\omega)\mathbf{\Phi}^{(xx)}(\omega)$ and the corresponding eigenvector is a vector of beamforming weights. This result has its consequences: 1) In order to obtain beamforming weights, speech and noise SCM estimates are required, favorably eliminating the need for explicit prediction of transfer functions by design. 2) A generalized eigenvalue problem solver is required in inference. In Section 6.4.3, we will discuss the complications when the GEV beamformer is part of a model that should be differentiable to optimize specific

criteria. 3) A magnitude of the beamforming vector can be arbitrary. Therefore, unlike MVDR, GEV can introduce speech distortion.

Now, let us analyze an interesting property of the GEV beamformer that sheds light on the similarities between GEV and MVDR. It will also lead to a technique for compensating for potential speech distortion. Without loss of generality, we will consider a reverberation-free environment, noting that generalization to reverberant enclosures is straightforward [Warsitz and Haeb-Umbach, 2007]. Under such constraint and using (3.44), we can rewrite the generalized eigenvalue problem in (3.50) as:

$$\begin{aligned} \phi^{(ss)}(\omega)\mathbf{d}(\omega,\mathbf{k})\mathbf{d}^H(\omega,\mathbf{k})\mathbf{w}(\omega) &= \lambda(\omega)\mathbf{\Phi}^{(vv)}(\omega)\mathbf{w}(\omega) \\ \left[\phi^{(ss)}(\omega)\mathbf{\Phi}^{(vv)^{-1}}(\omega)\mathbf{d}(\omega,\mathbf{k})\mathbf{d}^H(\omega,\mathbf{k})\right]\mathbf{w}(\omega) &= \lambda(\omega)\mathbf{w}(\omega). \end{aligned} \tag{3.51}$$

We observe that the rank of $\mathbf{\Phi}^{(vv)^{-1}}(\omega)\mathbf{d}(\omega,\mathbf{k})\mathbf{d}^H(\omega,\mathbf{k})$ is one; therefore, there is only one eigenvector (and thus beamforming vector) with a non-zero eigenvalue. It is given as [Warsitz et al., 2008]:

$$\mathbf{w}(\omega) = \zeta(\omega)\mathbf{\Phi}^{(vv)^{-1}}(\omega)\mathbf{d}(\omega,\mathbf{k}), \tag{3.52}$$

where $\zeta(\omega)$ is a complex constant. By using (3.52) in (3.51), we get:

$$\begin{aligned} \mathbf{\Phi}^{(vv)^{-1}}(\omega)\phi^{(ss)}(\omega)\mathbf{d}(\omega,\mathbf{k})\mathbf{d}^H(\omega,\mathbf{k})\left[\zeta(\omega)\mathbf{\Phi}^{(vv)^{-1}}(\omega)\mathbf{d}(\omega,\mathbf{k})\right] &= \lambda(\omega)\left[\zeta(\omega)\mathbf{\Phi}^{(vv)^{-1}}(\omega)\mathbf{d}(\omega,\mathbf{k})\right] \\ \mathbf{d}(\omega,\mathbf{k})\left[\phi^{(ss)}(\omega)\mathbf{d}^H(\omega,\mathbf{k})\mathbf{\Phi}^{(vv)^{-1}}(\omega)\mathbf{d}(\omega,\mathbf{k})\right] &= \lambda(\omega)\mathbf{d}(\omega,\mathbf{k}). \end{aligned} \tag{3.53}$$

It provides us with an expression for the eigenvalue (i.e., SNR):

$$\lambda(\omega) = \phi^{(ss)}(\omega)\mathbf{d}^H(\omega,\mathbf{k})\mathbf{\Phi}^{(vv)^{-1}}(\omega)\mathbf{d}(\omega,\mathbf{k}). \tag{3.54}$$

Having confirmed that (3.52) is a true eigenvector of the generalized eigenvalue problem, we proceed by comparing it with the result for the MVDR beamformer in (3.40). Noting that the numerator is the same, we can conclude that beamforming weights of GEV and MVDR in the reverberation-free environment are equal up to a complex scalar[6] $c(\omega)$. Distinguishing beamforming vectors by superscripts, we can summarize that

$$\mathbf{w}^{(\mathrm{MVDR})}(\omega) = c(\omega)\mathbf{w}^{(\mathrm{GEV})}(\omega), \tag{3.55}$$

with $\mathbf{w}_{\mathrm{GEV}}(\omega)$ representing the principal eigenvector of the generalized eigenvalue problem in (3.50).

Knowing the relationship between the GEV and MVDR beamformers, we will describe how to eliminate speech distortions potentially introduced by the GEV at least partially. A constant term introduced in (3.55) can be seen as a single channel *post-filter* applied to the GEV beamformer output. Since we know that the MVDR satisfies the distortionless constraint, setting the post-filter to

$$c(\omega) = \frac{1}{\zeta(\omega)\mathbf{d}^H(\omega,\mathbf{k})\mathbf{\Phi}^{(vv)^{-1}}(\omega)\mathbf{d}(\omega,\mathbf{k})} \tag{3.56}$$

[6] We note that in reverberant enclosures, the equality up to a complex scalar holds for GEV and generalized MVDR (GMVDR) [Warsitz and Haeb-Umbach, 2007], a simple extension of MVDR aiming to recover the original source (as opposed to its reverberant image at the reference microphone).

would compensate for speech distortion since the MVDR formula in (3.40) would be recovered. We would also need to estimate the steering vector $\mathbf{d}(\omega, \mathbf{k})$ on top of the second-order statistics, which is not practical. Therefore, the aim is to compute $c(\omega)$ using available quantities. First, let us deal with $\mathbf{d}(\omega, \mathbf{k})$. Using (3.52), we can express it as $\mathbf{d}(\omega, \mathbf{k}) = \zeta^{-1}(\omega)\mathbf{\Phi}^{(vv)}(\omega)\mathbf{w}_{\mathrm{GEV}}(\omega)$. Plugging the expression into (3.56) yields

$$c(\omega) = \frac{\zeta^*(\omega)}{\mathbf{w}_{\mathrm{GEV}}^{\mathsf{H}}(\omega)\mathbf{\Phi}^{(vv)}(\omega)\mathbf{w}_{\mathrm{GEV}}(\omega)}. \tag{3.57}$$

The last unknown is the complex constant $\zeta(\omega)$. We will seek for an approximation via $|\zeta(\omega)|^2$, where we will make use of (3.52), giving $\|\mathbf{\Phi}^{(vv)}(\omega)\mathbf{w}_{\mathrm{GEV}}(\omega)\|_2^2 = \|\zeta(\omega)\mathbf{d}(\omega, \mathbf{k})\|_2^2$. Since $\|\mathbf{d}(\omega, \mathbf{k})\|_2^2 = N$[7], then

$$|\zeta(\omega)|^2 = \frac{\mathbf{w}_{\mathrm{GEV}}^{\mathsf{H}}(\omega)\mathbf{\Phi}^{(vv)}(\omega)\mathbf{\Phi}^{(vv)}(\omega)\mathbf{w}_{\mathrm{GEV}}(\omega)}{N}. \tag{3.58}$$

By neglecting phase and using only magnitude of $\zeta(\omega)$, we eventually get

$$c_{\mathrm{BAN}}(\omega) = \frac{\sqrt{\mathbf{w}_{\mathrm{GEV}}^{\mathsf{H}}(\omega)\mathbf{\Phi}^{(vv)}(\omega)\mathbf{\Phi}^{(vv)}(\omega)\mathbf{w}_{\mathrm{GEV}}(\omega)/N}}{\mathbf{w}_{\mathrm{GEV}}^{\mathsf{H}}(\omega)\mathbf{\Phi}^{(vv)}(\omega)\mathbf{w}_{\mathrm{GEV}}(\omega)}, \tag{3.59}$$

which is called *blind analytical normalization* (BAN) post-filter [Warsitz and Haeb-Umbach, 2007]. Since an estimate of $\mathbf{\Phi}^{(vv)}(\omega)$ is required for the GEV beamforming weights computation, the BAN post-filter computation is straightforward. We observed that certain approximations have been made during the derivation. We also note that in practice, the quantities used to compute the post-filter are just estimates (that do not necessarily meet the theoretical assumptions). Therefore, the GEV beamformer with the BAN post-filter will not match exactly the MVDR in real-world applications.

## 3.3 Broadband Beamforming

In previous sections, we assumed the processing of a single frequency using the DTFT. Noting that the DTFT spectrum is continuous and speech signals are inherently broadband, we can conclude that beamforming would not be tractable in practical applications. However, we still want to benefit from beamforming in speech processing. Approaches allowing for spatial filtering for (not only) speech sources are called *response-invariant* or *broadband beamforming* [Benesty et al., 2008].

A practical and widely adopted approach decomposes broadband signals to narrowband ones and estimates beamforming weights for each band separately [Trees, 2002, Benesty et al., 2008]. Specifically, we aim to beamform discrete signals (obtained by sampling the continuous ones) employing discrete Fourier transform (DFT) for converting them into their frequency-domain representations. Each output sample of the DFT corresponds to a specific frequency band, commonly referred to as a *bin*. When the DFT is applied to short, windowed segments (frames) of the signal, the process is known as the short-time Fourier transform (STFT).

[7] We note that in a general case of reverberant environment, when $\mathbf{d}(\omega, \mathbf{k})$ is substituted by the actual transfer function $\mathbf{g}(\omega)$, the approximation $\|\mathbf{g}(\omega)\|_2^2 \approx N$ is still used in order to get an approximate post-filter [Warsitz and Haeb-Umbach, 2007].

To summarize, all discrete microphone signals are converted to the STFT domain. Then, we estimate beamforming weights independently for each frequency bin. Beamforming is performed in each bin by combining channels with the estimated weights to yield a single-channel frequency-dependent output. Finally, inverse STFT converts per-bin outputs to the time domain. The process for one frame and one bin is presented in Figure 3.13.

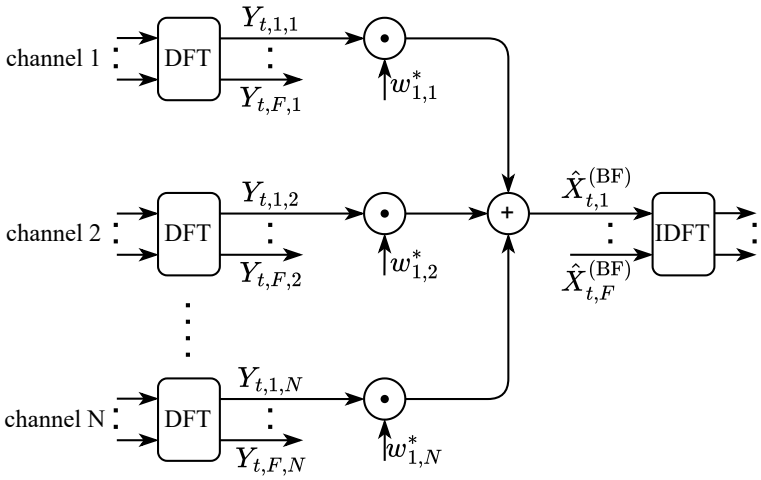



Figure 3.13: Scheme for broadband beamforming. We present computation for a single frame $t$ and frequency bin $f = 1$. The channels are combined by the beamforming vector $\mathbf{w}_f = [w_{f,1}, w_{f,2}, \ldots, w_{f,N}]$. Symbol $(\cdot)^*$ represents the complex conjugate.

### 3.3.1 Signal Model

Making an analogy with the signal model defined in Section 3.2.1, we will present an alternative model for broadband beamforming that we will use throughout our experiments. Let $\mathbf{s} \in \mathbb{R}^{\mathsf{T}}$ be a time-domain discrete speech signal, where $\mathsf{T}$ denotes the number of samples, and $\mathbf{S} \in \mathbb{C}^{T \times F}$ be its STFT-domain image with $T$ representing frames and $F$ frequency bins. Let us consider the same assumptions as in the previous model: the microphone array has $N$ sensors (channels) and noise is additive. Denoting the image of noise at the $n$th microphone as $\mathbf{V}_n \in \mathbb{C}^{T \times F}$, we assume the mixture captured by the $n$th sensor to be

$$\begin{aligned} \mathbf{Y}_n &= \mathbf{X}_n + \mathbf{V}_n \\ &= \mathbf{S}\left[\mathrm{diag}\left(\mathbf{g}_n\right)\right] + \mathbf{V}_n, \quad n = 1, \ldots, N, \end{aligned} \tag{3.60}$$

where $\mathbf{g}_n \in \mathbb{C}^F$ is the relative transfer function (RTF) between the speech source and the $n$th microphone, and the $\mathrm{diag}(\cdot)$ operator constructs a diagonal matrix with the diagonal comprising elements of its argument. Then, $\mathbf{X}_n \in \mathbb{C}^{T \times F}$ is a reverberant image of speech at the microphone $n$.

Let us consider that $\mathbf{Y} \in \mathbb{C}^{T \times F \times N}$ is an order-3 tensor grouping observations $\mathbf{Y}_n, n = 1, \ldots, N$. Then, each component $Y_{t,f,n} \in \mathbb{C}$ of $\mathbf{Y}$ can be interpreted as a complex value of a T-F bin (with indices $t$ and $f$) of the $n$th channel. As in Section 3.2.1, it will be helpful to arrange channels in vectors:

$$\mathbf{y}_{t,f} = [Y_{t,f,1}, Y_{t,f,2}, \ldots, Y_{t,f,N}]^{\intercal}. \tag{3.61}$$

Vectors $\mathbf{x}_{t,f}$ and $\mathbf{v}_{t,f}$ can be constructed analogically. Having equipped ourselves with the notation, we can formulate the beamforming operation respecting the outlined decomposition of the broadband signals as:

$$\hat{X}_{t,f}^{(\mathrm{BF})} = \mathbf{w}_f^{\mathsf{H}} \mathbf{y}_{t,f}, \tag{3.62}$$

where $\mathbf{w}_f \in \mathbb{C}^N$ represents an $N$-dimensional vector of beamforming weights. Note that we consider a time-independent beamformer. Analyzing the formula, we can conclude an interesting fact. Since the weights are applied by multiplication followed by summation, the operation equals filtering followed by summation of the resulting signals in the time domain. Therefore, such an operation is known as *filter-and-sum* beamforming.

With the defined signal model, it is straightforward to adapt formulas for the previously defined MVDR and GEV beamformers by rewriting symbols. However, we stress that second-order statistics are now estimated as expectations over frames. In order to give an example, the MVDR formula in (3.46) can be rewritten as:

$$\mathbf{w}_f = \frac{\left(\mathbf{\Phi}_f^{(vv)}\right)^{-1} \mathbf{\Phi}_f^{(xx)}}{\mathrm{tr}\left\{\left(\mathbf{\Phi}_f^{(vv)}\right)^{-1} \mathbf{\Phi}_f^{(xx)}\right\}} \mathbf{u}, \tag{3.63}$$

where $\mathbf{\Phi}_f^{(xx)}$ and $\mathbf{\Phi}_f^{(vv)}$ represent per-frequency spatial covariance matrices of (reverberant) speech and noise, respectively.

# Chapter 4

# Corpora

Data is a cornerstone of every machine learning application. While initiatives in various fields supplied considerable amounts of data for research, multi-channel speaker verification has suffered from a lack of data for an extended period. We can identify multiple reasons: 1) Multi-channel SV can be considered a niche field with a smaller community. Then, extensive data collection, which is vastly resource-demanding in the case of microphone arrays, is hardly justifiable. 2) Numerous other fields have benefited from scraping data from the internet. While it is tractable for single-channel scenarios, searching for and obtaining audio from simultaneously recording microphones with speaker labels is extremely difficult. Such issues are particularly pressing because of a cyclic dependency: if there is not enough reliable data, interest from the research community is limited; if the interest is limited, there is not enough push for collecting sufficient corpora.

Hindered by the lack of datasets, one of the goals of our work was to improve the data situation in multi-channel SV. In this chapter, we will describe our contribution to the community, considering data. First, we will outline the BUT ReverbDB dataset [Szőke et al., 2019], which is useful both for SV and ASR. Next, we will be concerned with our steps toward creating and publishing MultiSV [Mošner et al., 2022b] and MultiSV2 [Mošner et al., 2025] datasets. However, before delving into the details of the data preparation process, we will summarize the datasets commonly used in the single-channel SV, VoxCeleb versions 1 and 2 [Nagrani et al., 2017, Chung et al., 2018], which were both instrumental for MultiSV and for developing our systems.

## 4.1 VoxCeleb

The need for a substantial amount of training data naturally accompanied the rise of embedding extractors employing neural networks. The VoxCeleb datasets [Nagrani et al., 2017, Chung et al., 2018] represent a significant milestone in developing SV systems since they have satisfied the training data demand in academia for many years. It is reflected in a vast body of research publications and in numerous VoxSRC challenges[1] using the data [Huh et al., 2024]. Not only did VoxCeleb provide a vast volume of data for training and evaluation, but the authors also came up with an innovative pipeline to scrape data from the internet that inspired recent data collections [Lin et al., 2024b, Lin et al., 2024a].

[1] List of VoxSRC challenges with descriptions is provided at https://www.robots.ox.ac.uk/~vgg/data/voxceleb/. The challenges were described in the following papers: VoxSRC 2019 [Chung et al., 2019], VoxSRC 2020 [Nagrani et al., 2020a], VoxSRC 2021 [Brown et al., 2022], and VoxSRC 2022 [Huh et al., 2023].

The data was released in two versions: VoxCeleb1 [Nagrani et al., 2017] and VoxCeleb2 [Chung et al., 2018]. The corresponding data-collecting algorithms employed a similar scraping pipeline, which we will briefly overview. We do not aim to provide an extensive description with details specific to the two versions, as it is out of the scope of this text. However, we will cover a high-level structure and refer the interested reader to the respective publications for more details.

- The pipeline starts by preparing a list of candidate identities. They are selected from a face recognition dataset containing actors, sportspeople, or entrepreneurs.
- In the second stage, videos of the identities are searched for on YouTube. Each identity name is appended with the word „interview“ to increase the chance of downloading a video with the person speaking rather than engaging in a sport or music show.
- Face detection is performed for each frame after obtaining the videos. The detected faces are eventually grouped using position-based tracking.
- In order to discard clips with dubbing and increase the likelihood that the voice corresponds to the obtained face sequence, the next step performs active speaker verification. The main idea is to determine a correlation between speech and video.
- Eventually, the decision on whether the candidate's identity corresponds to the face track is made. To this end, the images are passed to an image classifier trained on the dataset from the first stage. Comparison of a classification score with a high threshold provides a final verdict on keeping the example in the final dataset.

Other steps might be involved depending on the version, such as removing duplicates and obtaining nationality labels. VoxCeleb1 and VoxCeleb2 represent standalone datasets for building single-channel SV systems as they provide development and test data. However, combining them to provide a more reliable system evaluation and comparison is a common practice in the research community. First, we will summarize statistics of the individual corpora, and then, we will summarize verification lists, stressing the combination of the datasets.

**VoxCeleb1** VoxCeleb1 was the pioneering dataset. Its development and test parts comprise 1,211 and 40 speakers, respectively. The total number of utterances in the development set is 148,642. The test set contains many fewer utterances (4,874) due to a limited number of speakers.

**VoxCeleb2** Overall, VoxCeleb2 is a scaled-up dataset. Therefore, it was warmly adopted by the community. The development set provides as many as 1M utterances from 5,994 speakers. The total number of speakers and utterances in the test set is 118 and 36,237, respectively.

VoxCeleb provides three lists of verification trials [Nagrani et al., 2020b], whose summary is provided in Table 4.1. They differ in size and difficulty, but they also specifically limit training data:

**VoxCeleb1-O** The *original* trial list using utterances of the VoxCeleb1 test speakers. It is a relatively small and easy test list for today's standards. Current state-of-the-art models achieve EER substantially less than 1%. Therefore, the merit of reporting the VoxCeleb1-O results becomes questionable.

Table 4.1: VoxCeleb trial lists statistics. Note that we consider only the cleaned versions of the lists.

| Trial list | Speakers | Utterances | Verification trials |
|---|---|---|---|
| VoxCeleb1-O | 40 | 4,708 | 37,611 |
| VoxCeleb1-E | 1,251 | 145,160 | 579,818 |
| VoxCeleb1-H | 1,190 | 137,924 | 550,894 |

**VoxCeleb1-E** The *extended* version of the trial list. Compared to VoxCeleb1-O, it represents a more extensive and more representative trial list. It comprises utterances of all the VoxCeleb1 speakers. Therefore, training models on the VoxCeleb1 development data is prohibited when reporting results using this trial list.

**VoxCeleb1-H** Similarly to VoxCeleb1-E, it tries to use as many VoxCeleb1 speakers as possible, given a constraint that makes it *hard*. Therefore, it also limits training data as VoxCeleb1-E does. The constraint mentioned above dictates that trials contain utterances of speakers of the same gender and nationality.

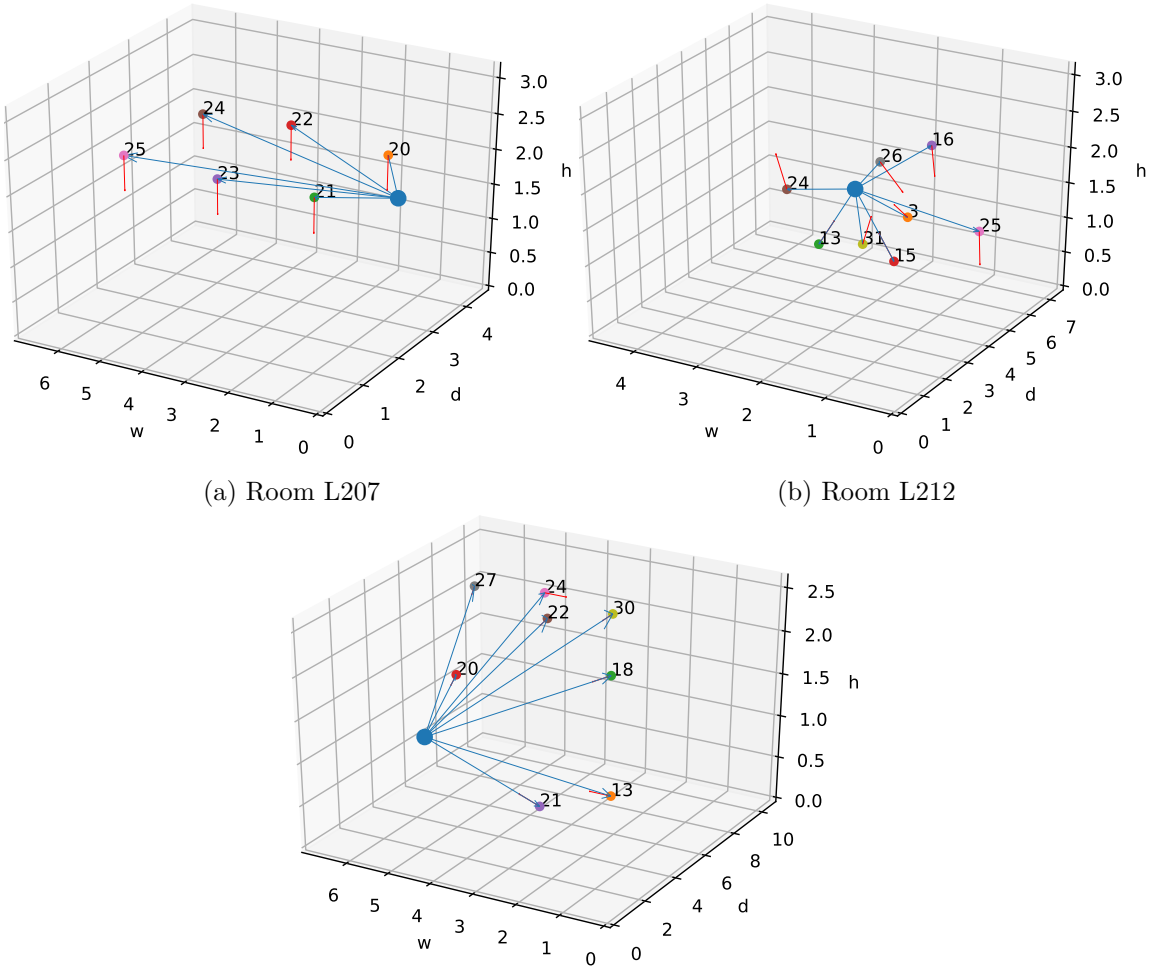
The consequence of the discussion about the trial lists is that VoxCeleb1-E and VoxCeleb1-H trial lists cannot be used for evaluation when training systems on the VoxCeleb1 development data. On the other hand, it is possible to measure verification metrics using all of the trial lists when training models on VoxCeleb2 dev. Since this combination of datasets provides a more reliable and complete scheme, it has become widely embraced.

## 4.2 Retransmitted BUT RevertbDB Corpus

The lack of data and crude approximations imposed by fast simulation algorithms motivated our initial data-related work, which dates back to 2019 [Szőke et al., 2019], in which we built a dataset of far-field speech, room impulse responses (RIRs), and ambient noises, called *BUT ReverbDB*[2]. It aimed to foster both ASR and SV in far-field conditions as it comprises data based on LibriSpeech Test-clean and part of the NIST Speaker Recognition Evaluation 2010 dataset [Martin and Greenberg, 2010]. We employed an alternative that fills a qualitative gap between unsatisfying simulation (despite the improvement of realism) and resource-demanding real speaker recording: *retransmission*, i.e., replaying speech recordings with a loudspeaker and capturing them with microphones again.

Since variability is an important aspect of evaluation data, recording sessions were held in eight acoustically diverse rooms of different sizes and RT60 parameters, ranging from 0.59 to 1.85 s. Although RIRs were measured (via exponential sine sweep) in all of them, speech data were re-recorded in five rooms only. For our dataset, we retransmitted recordings using a high-quality loudspeaker. Each session was captured by 31 simultaneously recording microphones. Importantly, the data is sample-synchronous across all microphones. This ensures the correct time alignment of signals in different channels. Given the setup of the microphone positions, we conducted multiple recording sessions in one room with different loudspeaker positions.

[2] https://speech.fit.vutbr.cz/software/but-speech-fit-reverb-database




(a) Room L207

(b) Room L212

(c) Room Q301

Figure 4.1: Rooms from the BUT ReverbDB corpus used in our experiments. Room dimensions are in meters; $w$, $d$, and $h$ stand for width, depth, and height, respectively. The blue circle represents a loudspeaker, color dots denote microphones (with their indices according to BUT RverbDB), and red arrows directions of microphones.

Collecting such a dataset is a complex task in terms of hardware and software. In [Szőke et al., 2019], not only did we describe the dataset, but the publication also serves as a cookbook for practitioners building similar corpora.

### Speaker Verification Evaluation Data

As noted before, the source audio was taken from a subset of NIST SRE 2010 data [Martin and Greenberg, 2010]. Although users of BUT ReverbDB are free to use speech data from all five rooms, loudspeaker settings, and microphones, we limited ourselves to a subset to be able to conduct controlled experiments, keeping the analysis manageable. We prepared the SV evaluation data that encompassed reverberant and spatially diverse environments. We selected three distinct rooms with different acoustic profiles and reverberation times

(RT30[3]) of 0.75 s, 0.53 s, and 0.65 s (see Figure 4.1). These values represent moderately reverberant environments, such as those encountered in home, office, or meeting-room settings. Out of 31 microphones that recorded simultaneously, we selected 6 or 8 microphones arbitrarily placed, depending on the room. These can be thought of as ad-hoc microphone arrays without particular geometry. We show them along the loudspeaker in Figure 4.1.

The evaluation speaker verification dataset comprises a total of 932 retransmitted speech recordings. Out of these, 459 contain female voices, and 473 contain male voices. The recordings include utterances from 300 unique speakers, with an equal gender distribution of 150 male and 150 female speakers. The average duration of recordings is approximately 38 s.

## 4.3 MultiSV Dataset

Motivated by the unconsolidated data situation and the lack of a standard benchmark in the field, we strived to design a comprehensive corpus for training and evaluating text-independent multi-channel speaker verification systems. We believe that the availability of a standard corpus may foster research in the field, allow for a fair comparison of techniques, and identify promising research directions.

In this section, we describe the data situation when creating our corpus and the steps toward achieving the goal we set. Eventually, our efforts resulted in publishing multi-channel SV training, development, and evaluation datasets, MultiSV[4] [Mošner et al., 2022b], a corpus based on publicly available data, and its extension, MultiSV2 [Mošner et al., 2025]. Apart from multi-channel SV, MultiSV is also readily available for dereverberation, denoising, and speech enhancement experiments.

### 4.3.1 Data Situation Before MultiSV

Previous studies on multi-channel SV often adopted some single-channel corpus (such as the dataset by SiTEC used in [Yang and Chang, 2019] or AISHELL-ASR0009-[ZH-CN] in [Cai et al., 2019]) with examples created by room simulation. In some studies [Taherian et al., 2019, Taherian et al., 2020], publicly available, albeit non-free datasets, such as single-channel NIST SRE 2008 data, were utilized for training. The mentioned inconsistency points out the lack of common multi-microphone data for SV experiments and difficulties with reproducibility and comparison.

Over the years, a few data-collecting or assembling activities have emerged, which we will briefly summarize. One of them resulted in the HI-MIA dataset [Qin et al., 2020]. It is unique in that it comprises recordings of 340 genuine speakers (not retransmitted speech). Each speaker uttered 160 (English and Chinese) wake words and was recorded by six 16-channel microphone arrays and one close-talking microphone. The corpus contains recordings both with and without background noise. It is composed of two sub-datasets: AISHELL-wakeup and AISHELL-2019B-eval. The former was used in [Qin et al., 2020] for fine-tuning and the latter for evaluation. AISHELL-wakeup was not used for embedding

[3]Reverberation time RT60 represents the time needed for a sound level in a room to decay by 60 dB after the source has been turned off. The measurement of RT60 is usually not easy since 60 dB represents a big difference between sound levels. Therefore, alternative metrics such as RT30 are more practical. Then, the time required for a sound level to decrease by 30 dB is multiplied by 2 to obtain an approximation of RT60.

[4]We released source data lists, instructions, and scripts for preparation at https://github.com/BUTSpeechFIT/MultiSV.

extractor training from scratch, as it contains only 16 hours per microphone (which is insufficient for current models). As suggested by the wake word *Hi Mia* in the name, the dataset was designed for *text-dependent* SV only.

A noteworthy challenge that dealt with far-field *text-independent* SV was Task 2 of the Far-Field Speaker Verification Challenge (FFSVC) 2020[5] [Qin et al., 2020]. It employed Mandarin data and provided a training set comprising microphone array recordings of 120 speakers. For evaluation, the enrollment segments were recorded with a cell phone. Test segments were captured with microphone arrays (of 4 channels).

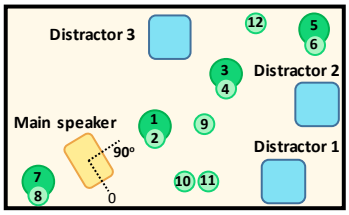
Libri-adhoc40 [Guan et al., 2021] is a collection of retransmitted and precisely synchronized audio. It provides reverberant speech segments of 331 speakers from the LibriSpeech corpus [Panayotov et al., 2015]. Uniquely, 40 equal microphones were used for collection, and each microphone recorded 110 hours of speech. The placement of microphones and speakers for the training part (train-clean-100) differed from the settings for the development/evaluation part (dev-clean/test-clean). The collection took place in a large reverberant room (with an RT60 of 900 ms). For reference, the same data was also recorded in an anechoic chamber. Compared to HI-MIA, no additive noise was replayed. Although [Guan et al., 2021] presents ASR results, the dataset is also usable for text-independent SV as demonstrated in [Liang et al., 2021].

Some works tried to reuse existing data to prepare SV trials. CHiME-5 challenge [Barker et al., 2018] surprised the community by the great difficulty of the real data collected during numerous dinner parties of friends. The appeal of this data led to the CHiME-5 speaker recognition benchmark [Garcia-Romero et al., 2019]. It reused recordings from 18 (out of 20) parties of 39 speakers. It aimed at providing means to compare various scenarios: single speaker vs. multiple speakers, close-talking vs. far-field microphones, and single microphone vs. microphone arrays. Unfortunately, despite its attractiveness, it has never been publicly released.

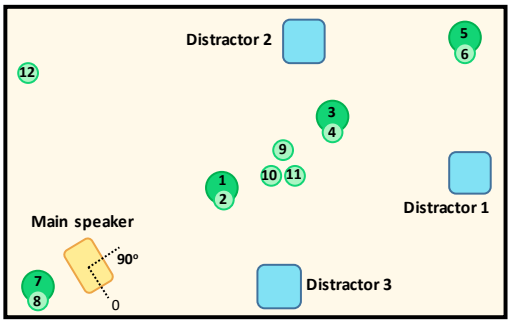
### 4.3.2 Analysis of the VOiCES Corpus

The VOiCES from a Distance Challenge 2019 [Nandwana et al., 2019] aimed at evaluating SV systems using single-channel trials based on the Voices Obscured in Complex Environmental Settings (VOiCES) corpus [Richey et al., 2018]. The acoustically challenging dataset, released by SRI International and Lab41 of In-Q-Tel, represents a tradeoff between simulation and far-field recordings of genuine speakers in rooms: A foreground speech was replayed by a loudspeaker on a motorized rotating stand (to simulate head movements) and recorded by 12 microphones. This procedure is usually referred to as *retransmission* (see also Section 4.2 on our BUT ReverbDB). The foreground (read) speech, totaling 15 hours, was selected from the clean part of LibriSpeech [Panayotov et al., 2015]. Along with the foreground speech, a noise was simultaneously replayed by different loudspeakers. To simulate a real-world background, the authors chose to perform recording under four noise conditions: television, radio, babble, and none (ambient room noise). The television and radio noises were replayed with one corresponding loudspeaker, while the babble noise was produced by replaying speech with three loudspeakers. A set of television noise samples contains audio excerpts from publicly available movies and shows[6]. The music noise was drawn from the MUSAN dataset [Snyder et al., 2015]. Random recordings from the „us-gov“ part of MUSAN served as a base for babble noise generation: speech of three random

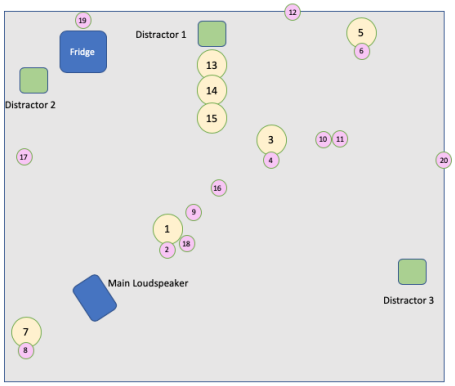
[5] http://2020.ffsvc.org/
[6] https://publicdomainmovie.net/



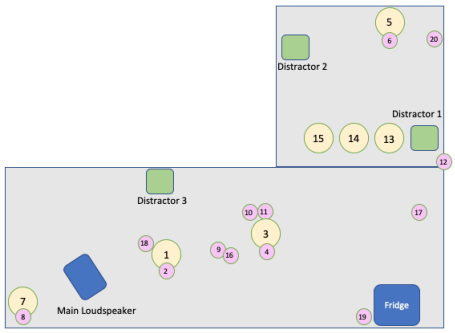

(a) Room 1 (3.71 m × 2.72 m).




(b) Room 2 (5.72 m × 4.01 m).




(c) Room 3 (7.62 m × 7.62 m).




(d) Room 4 (9.75 m × 4.88 m + 3.05 m × 3.05 m).

Figure 4.2: Rooms where recording sessions of the VOiCES corpus took place, along with positions of microphones and loudspeakers. The main loudspeaker replayed a foreground speech; the distractor loudspeakers were dedicated to noise. Big circles represent dynamic studio-quality microphones (SHURE SM58); small circles denote condenser lavalier (AKG 417L) or dynamic lavalier (SHURE SM11) microphones. The images are reprinted from the official website[7].

speakers was dedicated to each of the three distractor loudspeakers, resulting in a mixture of nine speakers' speech. The recording sessions took place in four distinct furnished rooms with different acoustic profiles, shown in Figure 4.2.

We aimed to design a testbed for multi-channel SV using freely available public-domain data so that the paywall is not a limiting factor for adopting it. The VOiCES dataset turned out to be particularly beneficial for achieving our goal for the following reasons:

- As outlined before, the dataset comprises recordings of the same utterances concurrently captured by multiple microphones in the same environments. It then allows for grouping sensors into ad-hoc microphone arrays.

- The data was retransmitted, making it more realistic compared to the simulation. Although simulated data is used, especially in speech separation and enhancement, experimental evidence shows that techniques performing well in such scenarios may fall short when confronted with real data. A favorable aspect of retransmitted data is that reverberation is genuine. In data simulation, reverberation is commonly achieved by convolving speech and noise signals with artificially generated room impulse re-

[7] https://iqtlabs.github.io/voices/rooms/

Table 4.2: Statistics of the development and evaluation trial lists defined for the VOiCES challenge [Nandwana et al., 2019]. The term *clean* in the enrollment recordings section of the evaluation set denotes the fact that also original (not retransmitted) LibriSpeech utterances were used. Noise types are abbreviated as follows: B — babble, T — television, M — music, and N — none.

| | | **Enrollment** | **Test** | **Total** |
|---|---|---|---|---|
| **Development set** | **env** | room 1 | room 2 | room 1, 2 |
| | **noise** | N | N, B, T, M | N, B, T, M |
| | **spkrs** | 103 | 189 | 196 |
| | **utts** | 128 | 489 | 617 |
| | **# mics** | 2 | 8 | 2 + 8 |
| | **files** | 256 | 15,648 | 15,904 |
| **Evaluation set** | **env** | clean; room 3 | room4 | clean; room 3,4 |
| | **noise** | —; N | N, B, T | N, B, T |
| | **spkrs** | 44; 56 | 96 | 100 |
| | **utts** | 186; 70 | 336 | 592 |
| | **# mics** | —; 2 | 11 | 2 + 11 |
| | **files** | 186; 140 | 11,066 | 11,392 |

sponses. They represent a significant drawback as they are usually generated by a ray abstraction of acoustics (e.g., image source method [Allen and Berkley, 1979, Diaz-Guerra et al., 2021], ray tracing) or even by its approximation [Luo and Gu, 2024], which hardly reflects reality. This is because simulating the wave nature of acoustics is very computationally expensive.

- The VOiCES from a Distance Challenge 2019 provided participants with development and evaluation verification trial lists. Not only are they helpful in assessing the performance of single-channel systems (that can be compared with multi-channel ones), but they can be used as a base when constructing new multi-channel trials.

Since we identified a suitable dataset serving as a source of audio and providing exemplar verification trials, we will focus on analyzing its properties in the following section. The outcome will shed light on the utility of the data and eventually will allow us to design a multi-channel SV benchmark.

### Breakdown of the VOiCES Challenge Trial Definition

The overall statistics of the original data subsets are shown in Table 4.2. The development and evaluation sets share some properties: sets of enrollment and test utterances are disjoint and recorded in different rooms. The evaluation dataset is more difficult as enrollment recordings contain utterances retransmitted in room conditions and original clean speech of LibriSpeech (being more dissimilar to noisy, reverberant test audio). Clean enrollment recordings and those recorded in room 3 contain the voices of two disjoint sets of speakers. Sets of 196 speakers in the development portion and 100 speakers in the evaluation part are also disjoint.

Both development and evaluation trials were created such that, for every enrollment recording, multiple test files containing the same content (utterance, speaker, background

noise, and room) recorded with multiple microphones exist. Based on this property, test utterances will be assumed to come from arbitrary microphone arrays, resulting in a need for multi-channel processing. In the following text, the analysis will focus only on the test part of the trials (suitable for forming ad-hoc microphone arrays).

#### Development Test Set

In the original set of test recordings, each utterance is uttered by only one speaker. Each speaker uttered at least one utterance and at most 6 utterances; the average is 2.6 utterances per speaker. Every utterance was recorded in all noisy conditions — music, babble, television — and also *without* the presence of a distractor (*none* in Table 4.2). Eight microphones — 2, 4, 6, 8, 9, 10, 11, and 12 — were recording the speech simultaneously. Positions and types of individual microphones are shown in Figure 4.2 and detailed in [Richey et al., 2018].

#### Evaluation Test Set

Similar to the development test recordings, every utterance is uttered by only one speaker. The minimum of utterances per speaker is 1, the maximum is 9, and the average is 3.5. Every utterance was recorded with babble, a television distractor in the background, or without any noise (note that the music noise was missing). Eleven microphones — 4, 6, 8, 9, 10, 11, 12, 16, 17, 18, and 19 — were recording simultaneously.

### 4.3.3 MultiSV Development and Evaluation Data

MultiSV defines development and evaluation parts containing different lists of verification trials. The retransmitted development set (*dev retr*) is derived from the original VOiCES development set as follows: In the original VOiCES trial list, we identified groups of 8 trials sharing exactly the same enrollment recording (also recorded by the same microphone) and the same test utterance recorded by eight different microphones (in the same recording session). Each set of eight trials was reduced to two trials, sharing the enrollment utterance and comprising multi-channel test audio obtained by grouping four random microphones (sampled without replacement to cover all of them), forming ad-hoc microphone arrays. We note that MultiSV contains multiple protocols (detailed later). At this point, however, it is worth mentioning that the test multi-channel parts of trials created according to the described procedure appear in all protocols. The difference between them resides in the way enrollment recordings were obtained. The retransmitted evaluation set (*eval retr v1*) was constructed analogously from the original VOiCES evaluation set. However, 11 microphones captured the audio scenes in the case of test recordings. In order to keep the number of microphones in the created ad-hoc arrays consistent with the development set, we randomly selected two pairs of four microphones per given enrollment recording. The last array consists of three remaining microphones and one microphone randomly selected out of the already used ones.

For the evaluation part, the original signals were recorded by microphones placed across a large L-shaped room (room 4 as indicated in Table 4.2 and shown in Figure 4.2d). Sensors placed right next to the distractor speaker (mic. 12), fridge (mic. 19), or behind a corner far from the source speaker (mic. 6) are included as well. Signals of such microphones may have negative SNRs and be confusing for single-channel enhancement models: Poor predictions for such microphones in an array can lead to erroneous results (unless properly

handled by altered training data). Due to the described properties reflected in *eval retr v1*, we defined a second version (*eval retr v2*)[8]. The only difference compared to v1 is that the problematic microphones (6, 12, 19, and also 17) were replaced by those that are close to non-problematic ones (1, 2, 3, and 7), which resulted in more compact arrays.

As noted before, we devised multiple protocols to support various use cases. Although multi-channel test parts of the trials remain unchanged across protocols, we vary the properties of enrollment segments while keeping utterances unchanged. Therefore, we can get an exact mapping between trials in different protocols and study the influence of acoustics and the multiplicity of channels, irrespective of other effects. Based on the properties of the enrollment segments, we defined the following protocols:

- **SRE** (single-channel retransmitted enrollment): SRE is closest to the original VOiCES challenge trials since the enrollment recording for the multi-channel test side of a trial played the same role for all the constituent test segments in the original trial list. Consequently, the enrollment segments in the development part are retransmitted recordings. Those in the evaluation part can be clean or retransmitted, as shown in Table 4.2.
- **CE** (clean enrollment): Single-channel enrollment utterances are always clean. They were drawn from the LibriSpeech source audio.
- **MRE** (multi-channel retransmitted enrollment): Enrollment segments are 4-channel reverberant recordings from quiet rooms.
- **MRE_hard**: The MRE_hard protocol is similar to MRE but with the noise source playing in the background.

We also enriched previously described protocols with one that allows for more controlled experiments, the simulated one. The enrollment segments correspond to those used in SRE. The simulated sets of multi-channel test segments use the same source speech as the retransmitted sets. Therefore, the resulting development (*dev simu*) and evaluation (*eval simu*) definitions are equal to the retransmitted ones in terms of speakers, utterances, and the number of trials. The difference resides in reverberation and noise samples. We attempted to match the domain of the original noises; therefore, we used the following data:

- **Music** (4.2 h + 4.2 h): a subset of MUSAN music equally split between the development and evaluation sets,
- **MUSAN noises** (0.6 h + 0.6 h): 10% and 10% of the "noise" part of MUSAN,
- **Freesound.org and self-recorded noises** (1.8 h + 2.2 h): real fan, HVAC[9], shop, crowd, library, office, and dishes sounds.

In simulation, RT60 reverberation times of RIRs generated using the image source method [Allen and Berkley, 1979], ISM, were uniformly drawn from the interval [0.3, 0.9] s. Mixing SNRs were uniformly drawn from [3, 20] dB.

Overall, 4,005,888 original development trials were reduced to 1,001,472 trials. The verification list comprises 996,448 impostor trials and 5,024 target trials. A total of 3,607,516 original evaluation trials were reduced to 983,868 (with 9,939 target ones).

[8] We note that our default version of the evaluation set is *eval rert v1*. Therefore, whenever we do not specify the version in our experiments, we implicitly consider version *v1*.

[9] HVAC stands for heating, ventilation, and air conditioning.

Table 4.3: Per-microphone statistics of the MultiSV training set.

| | Training | Cross-validation |
|---|---|---|
| **Duration [h]** | 72.62 | 4.68 |
| **Speakers** | 500 + 500 | 45 + 45 |
| **Files** | 41,854 | 2,720 |

For multi-channel SV experiments, we suggest setting the prior probability of a target trial $P_{\text{tgt}}$ to 0.01 when reporting minDCF. It was originally used in the VOiCES challenge [Nandwana et al., 2019]. It also constituted an operating point providing statistically reliable results for our baselines in [Mošner et al., 2022b]. We observed that this operating point is reliable on the right side of the FA DR30[10] point, which marks the place where systems make just 30 false alarms. Setting an operating point close to or to the left of FA DR30 would produce unreliable results [Brümmer, 2010c].

### 4.3.4 MultiSV Training Data

Since collecting multi-channel training data is resource-demanding, we opted for a compromise — simulation. We decided to base the training data on VoxCeleb2 dev since it is a commonly used dataset for embedding extractors training.

The development partition of VoxCeleb2 comprises recordings with varying channel conditions and noise levels. As these recordings serve as source speech material for data simulation, clean signals are preferred. To ensure sufficient quality, we applied a preselection process based on signal-to-noise ratio (SNR) estimation, retaining only recordings with SNRs exceeding 20 dB. The final selection was divided into the training and cross-validation parts. The training part contains speech of 1,000 speakers with an even gender split (500 female and 500 male voices). For each speaker, the set contains at most 50 utterances. All recordings amount to 72.62 hours of audio. The cross-validation set comprises 90 speakers (45 females, 45 males), each represented by 35 utterances at most. The total duration of this part is 4.68 hours. The average duration of the utterances in both parts is 6.2 s. We show the statistics of the training set in Table 4.3.

To mimic indoor conditions, we performed a simulation considering a point source of speech and a point source of noise. Noises likely to occur indoors were selected from three sources:

- **Music** (66.3 h): FMA *small* dataset [Defferrard et al., 2017] where we excluded recordings that are present in the music part of MUSAN [Snyder et al., 2015] as these are used in the evaluation data,
- **MUSAN noises** (5.0 h): 80% of the "noise" part of the MUSAN dataset (i.e., without music and babble),
- **Freesound.org and self-recorded noises** (20.1 h): real fan, HVAC, shop, crowd, library, office, and dishes sounds.

In order to obtain reverberant speech and noise, we performed room impulse response (RIR) generation by image source method (ISM) [Allen and Berkley, 1979]. We simulated box-shaped rooms with 4-microphone ULAs. The length of ULAs ranged from 10 cm to 2 m.

[10]The name DR30 relates to Doddington's Rule of 30 [Doddington, 1998]. It suggests that at least 30 false alarms and 30 misses are required for a sensible system evaluation.

Table 4.4: Per-microphone statistics of the MultiSV2 training set.

| Attribute | Value |
|---|---|
| **Total duration [h]** | 1,157 |
| **Speech (after VAD) [h]** | 1,012 |
| **Female speakers** | 2,304 |
| **Male speakers** | 2,304 |
| **Files** | 512,604 |

Reverberation time RT60 was sampled uniformly from a range of [0.3, 0.9] s. The resulting RIRs were convolved with speech and noise sources. Mixing SNRs were uniformly drawn from [3, 20] dB.

For multi-channel SV models where a single-channel pre-training is possible (and/or required), we propose to pair MultiSV train with VoxCeleb 2 dev. Such a protocol allows fair comparison of different studies and also of our results as we follow it.

## 4.4 Extension of MultiSV — MultiSV2

Intending to support direct training of multi-channel embedding extractors (such as a joint model we present in Section 6.4.5 or a WavLM-based extractor in Section 7) whose training demands huge volumes of data, we aimed to extend our previous MultiSV training data. We presented a corpus which we termed MultiSV2 in [Mošner et al., 2025] and released a description for preparation in our MultiSV repository[11].

We base the extended multi-channel simulated training dataset also on a subset of VoxCeleb2 dev. The reason is twofold:

1. VoxCeleb2 dev is a de facto standard training corpus in the SV community. It is a large-scale corpus; therefore, it provides a wide enough set of recordings from which to select. Moreover, by choosing VoxCeleb2, we do not introduce overlap between training and test speakers of MultiSV.

2. Since the single-channel embedding extractors we developed in the context of the thesis were trained on VoxCeleb2, employing a subset of the original training speakers allows us to keep the model's classification layer.

When selecting speech data from VoxCeleb2, there are two contradictory requirements. First, since the recordings serve as a source of speech in spatial simulation, they should be clean. Second, a substantial volume of data is needed for speaker embedding extractors based on neural architectures. Our data selection relies on an SNR estimate; only recordings with an SNR higher than a threshold are preserved. As a tradeoff, we chose a threshold of 15 dB. The data list was eventually post-processed to obtain a gender-balanced set of speakers. The final selection comprises recordings of 2,304 male and 2,304 female speakers. The total number of files is 512,604, amounting to 1,157 hours of audio (1,012 hours of speech after VAD), as summarized in Table 4.4.

[11] https://github.com/BUTSpeechFIT/MultiSV/tree/main/MultiSV2

### 4.4.1 Noise Resources

For compatibility with MultiSV, each speech source is augmented with a noise source. Due to extensive speech data, we leveraged multiple corpora to obtain a comparable set of noise recordings in terms of duration: FMA large [Defferrard et al., 2017], WHAM! [Wichern et al., 2019], CHiME-3 [Barker et al., 2015], original MultiSV [Mošner et al., 2022b], and FSD50K [Fonseca et al., 2022]. We did not employ all the clips from the datasets, but we performed a selection described below.

Free Music Archive (FMA) is a large-scale collection suitable for music information retrieval. It provides high-quality tracks from 16,341 artists and 14,854 albums. It is distributed as FMA full, large, medium, and small. While the original MultiSV used the *small* subset, we opted for FMA *large* for MultiSV2. Similar to the full set, it contains all the tracks. The difference is that they are trimmed to 30 seconds. Despite the corpus comprising various metadata and the train/validation/test definition, we post-processed it in a way specific to our use case. Sets of MUSAN and FMA artists are not disjoint. Moreover, MultiSV contains MUSAN music in the development and evaluation parts. Therefore, we made sure to exclude the overlapping artists so that we could keep MultiSV for evaluation. Subsequently, we selected 734 hours of audio to preserve the ratio between music and non-music noises w.r.t. the original MultiSV train. We performed the selection in a round-robin way over artists, where we always picked a random song without replacement.

The WHAM! *noise dataset* is a collection of ambient noises from restaurants, cafes, bars, and parks. Paired with the WSJ0 corpus, it yields the WHAM dataset. In our scenario, we were interested only in the noise recordings. Since there is no clash with the evaluation data, we used all the recordings with 78.52 hours of audio.

Not only did the 3rd CHiME challenge provide a testbed for multi-channel automatic speech recognition, but it also supplied a corpus of real and simulated data. Noisy speech data and noises for simulation were recorded using a tablet equipped with six microphones. The recording sessions took place in public areas: cafes, street junctions, public transport, and pedestrian areas. When building our corpus, we were interested only in the provided noises. Out of six channels, we always selected only two to limit repetitions. In a given recording session, microphone 2 was always selected since it is the only one that faces backward. Out of the remaining ones facing forward, only one was randomly chosen. In total, CHiME-3 contributed 16.84 hours of audio.

FSD50K is a dataset comprising more than 51k clips from Freesound. While the data was labeled with 200 classes following the AudioSet Ontology[12], we manually selected 160 classes. We excluded recordings containing instruments (except for impact sounds, such as crashes) to avoid over-representation of music noises in our data. Another set of discarded classes contains human-vocalized clips, as they would cause source confusion. Since the development and evaluation sets of MultiSV are augmented with Freesound noises, we excluded overlapping files from FSD50K. Out of the total 100 hours of audio, our filtering steps left 75.86 hours.

We also included 25.1 hours of other Freesound and self-recorded clips from the original MultiSV training set to expand the noise corpus. In total, our noise data assembling activity supplied 930.32 hours of varied recordings.

In order to simulate multi-channel data, we generated RIRs for each speech-noise pair using the ISM. To mimic the evaluation data, we considered four microphones, one speech source, and one noise source randomly distributed around an imaginary box-shaped room,

[12] https://research.google.com/audioset/ontology/

where we imposed a margin of 0.4 m from walls and ceiling and a margin of 1.0 m from the floor. Reverberation time RT60 was uniformly sampled from the interval [0.2, 1] s. Wall sizes ranged from 2 to 10 m in one dimension and 2 to 11 m in another. A ceiling height was uniformly drawn from 2.3 to 5 m. Mixing SNR was eventually drawn from [3, 20] dB. We complement our description with a webpage providing further details along with necessary simulation parameters for each speech-noise pair to promote the reproducibility of the dataset and our experiments: https://github.com/BUTSpeechFIT/MultiSV/tree/main/MultiSV2.

# Chapter 5

# Speech Separation for Speaker Verification

We dedicate this chapter to speech separation, a pre-processing stage central to most multi-channel speaker verification models developed in this work. We focused on the analysis of known approaches as well as on the design and evaluation of our extensions to them. Consequently, the techniques presented here form a foundation for the multi-channel SV systems evaluated in the later chapter.

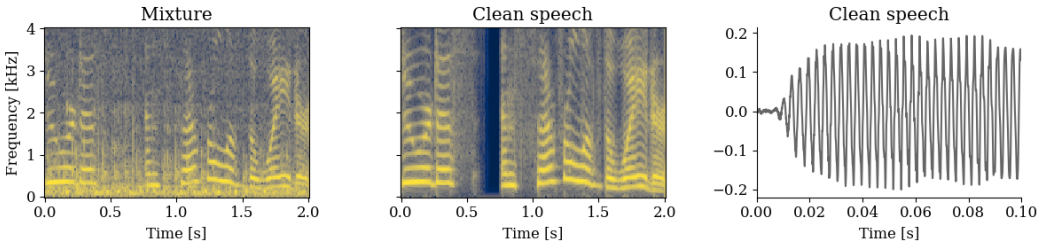
We first outline the evolution of single-channel and multi-channel separation methods that motivate our pre-processing models. We then analyze the role of hyperparameters in beamforming, focusing on the broadband MVDR as we use it throughout the chapter. What follows is a description of the architectures of neural network separation models that we use as backbones for our extensions. Importantly, we introduce the concept of maskless beamforming, as well as direct and indirect mask prediction for mask-based beamforming, combining the benefits of signal processing and the underlying speech separation networks. Finally, we introduce our approach to extending single-channel separation models to their multi-channel counterparts by combining cross-attention with beamforming, and present experimental results with the architecture.

## 5.1 Single-Channel Speech Separation Overview

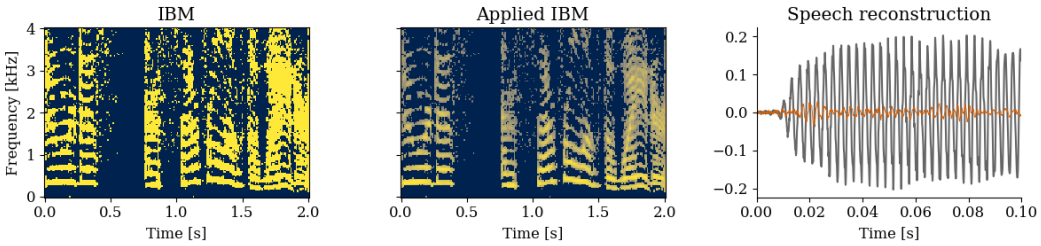
As defined in [Wang and Chen, 2018], *speech separation* is concerned with separating target speech from background interference. It can be divided into two special cases: *speech enhancement* and *speaker separation*. The former focuses on speech-nonspeech separation and the latter on segregating multi-talker speech. In speech enhancement, the nonspeech component may or may not be of interest, depending on the task. Due to close relatedness and mutual influence, we will discuss enhancement and separation, stressing that our experimentation draws inspiration from both.

### 5.1.1 Speech Enhancement

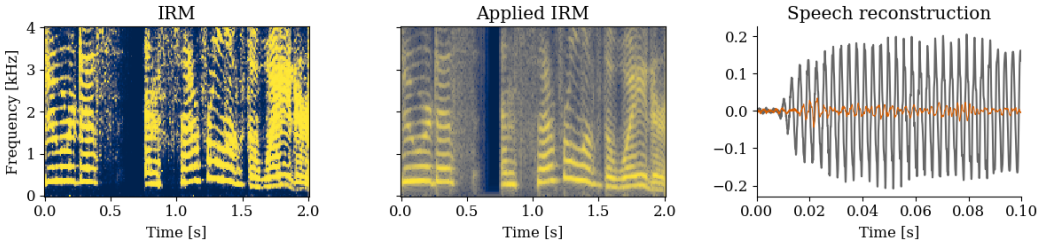
The early works formulated speech enhancement in the frequency domain. As an exceptional example, computational auditory scene analysis (CASA) [Wang and Brown, 2006], building upon perceptual principles, used the ideal binary mask (IBM) [Wang, 2005] as an estimation target. The IBM is a time-frequency (T-F) mask labeling each T-F bin with 1



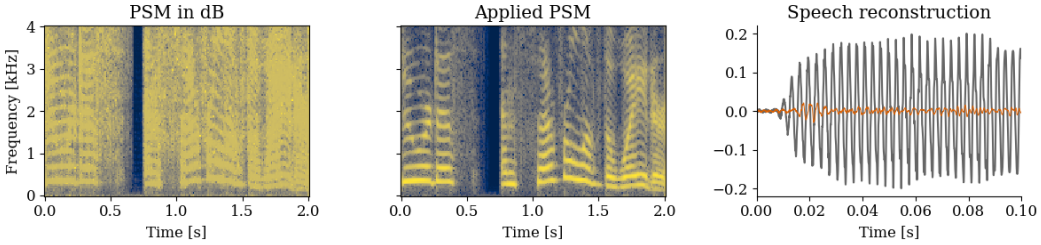

(a) Original mixture and speech to be reconstructed.



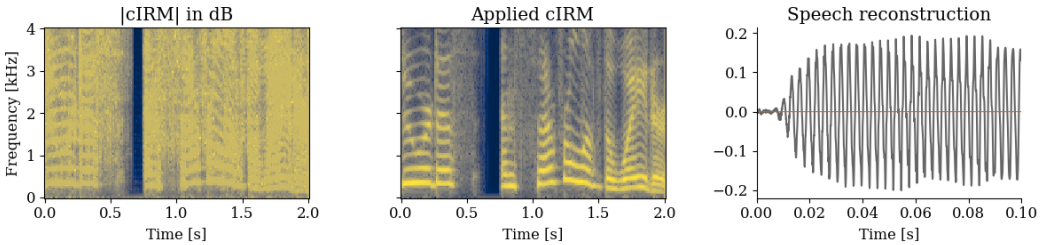

(b) Ideal binary mask (IBM).




(c) Ideal ratio mask (IRM).




(d) Phase sensitive mask (PSM).




(e) Complex ideal ratio mask (cIRM).

Figure 5.1: Demonstration of various T-F masks and their impact on the masked spectrum and reconstructed time-domain speech signal. The orange curve denotes an error signal.

or 0 based on the prevalence of speech or interference (see Figure 5.1b):

$$M_{t,f}^{(\text{IBM})} = \begin{cases} 1, & \text{SNR}(t,f) > \xi \\ 0, & \text{otherwise} \end{cases}, \tag{5.1}$$

where $\xi$ is a threshold. Noteworthy gains were achieved by treating speech enhancement as a binary T-F bin classification problem.

Inspired by the previous research, the pioneering studies published in 2012 [Wang and Wang, 2012a, Wang and Wang, 2012b] that introduced neural networks to speech enhancement took a classification approach in the frequency domain, resulting in *mask prediction*. Given an input obtained with a gammatone filterbank (cochleagram), per-subband neural networks (DNNs) were trained to predict binary labels. The DNNs eventually served as feature encoders. The last layer activations were used as inputs to structured perceptrons [Wang and Wang, 2012a] or conditional random fields [Wang and Wang, 2012b], providing the final IBM prediction. The studies were extended in [Wang and Wang, 2013] and paved the way for supervised frequency-domain DNN mask-based approaches. In addition to IBM, various other training targets have been explored [Wang et al., 2014]. Assigning a binary value to each T-F bin might seem crude. As an alternative, the ideal ratio mask (IRM) [Hummersone et al., 2014] takes real-valued numbers, acting as a soft version of the IBM (Figure 5.1c). The IRM T-F bins' values correspond to the following ratio:

$$M_{t,f}^{(\text{IRM})} = \left( \frac{|X_{t,f}|^2}{|X_{t,f}|^2 + |V_{t,f}|^2} \right)^{\beta}, \tag{5.2}$$

with $X_{t,f}, V_{t,f}$ representing T-F bins of speech and noise spectra following our signal model (Section 3.3.1) without the channel dimension, and with $\beta$ representing a tunable parameter usually set to 0.5 [Wang and Chen, 2018]. It follows from (5.2) that the IRM mask elements are bounded, taking real values between 0 and 1. Using IRMs leads to a regression (rather than classification) task where a common objective is the mean squared error (MSE) [Weninger et al., 2014]. As the names suggest, predicted IBMs or IRMs are typically element-wise multiplied with the magnitude spectrum of a mixture signal. Eventually, a time-domain signal is reconstructed using the mixture phase. Neglecting components' phases poses a bound on performance.

Aiming at the inclusion of phase information, [Erdogan et al., 2015] proposed a phase-sensitive mask (PSM). It is still a real-valued mask (as shown in Figure 5.1d) obtained by taking the real part of the ideal complex filter, leading to higher SNRs:

$$M_{t,f}^{(\text{PSM})} = \frac{|X_{t,f}|}{|Y_{t,f}|} \cos(\angle X_{t,f} - \angle Y_{t,f}). \tag{5.3}$$

A mask, theoretically capable of exactly recovering the target speech given an input mixture (see Figure 5.1e for confirmation), is termed a complex ideal ratio mask (cIRM) [Williamson et al., 2016]:

$$M_{t,f}^{(\text{cIRM})} = \frac{Y_{t,f}^{(r)} X_{t,f}^{(r)} + Y_{t,f}^{(j)} X_{t,f}^{(j)}}{\left(Y_{t,f}^{(r)}\right)^2 + \left(Y_{t,f}^{(j)}\right)^2} + j \frac{Y_{t,f}^{(r)} X_{t,f}^{(j)} - Y_{t,f}^{(j)} X_{t,f}^{(r)}}{\left(Y_{t,f}^{(r)}\right)^2 + \left(Y_{t,f}^{(j)}\right)^2}. \tag{5.4}$$

The superscripts $(\cdot)^{(r)}$ and $(\cdot)^{(j)}$ denote real and imaginary parts of complex numbers, respectively. Unlike other masks, cIRM is complex-valued, imposing requirements on the

network architecture. Its predictor has to estimate real and imaginary parts, which are preferred[1] over magnitude and phase. The issue with cIRM is its unbounded values. To combat associated optimization problems, values are usually constrained by a hyperbolic tangent during training [Williamson et al., 2016, Hao et al., 2021].

A direct optimization of masks does not necessarily optimize the final enhancement objective. An alternative approach, which still uses masks, is *signal approximation* [Weninger et al., 2014]. Models based on this scheme still internally predict masks. Instead of optimizing the mask with respect to the reference one, the mask is directly applied to the mixture even during training to yield speech estimates. Then, the loss function compares the estimated and ground-truth clean speech. Initial successful investigations encompassing real-valued masking [Weninger et al., 2014] facilitated research of complex-valued masks. Despite complex mask estimation with real-valued neural networks being feasible, the deep complex U-net (DCUnet) showed the benefits of using deep complex networks [Choi et al., 2019]. It laid the basis for its strong successors, such as DCCRN [Hu et al., 2020b].

As a complement to masking-based approaches, [Lu et al., 2013] introduced an alternative paradigm, *spectral mapping*, in 2013. While still defined in the frequency domain, the goal is to map an input mixture directly to its clean counterpart. Similar to early mask-based methods, initial mapping approaches estimated the magnitude spectrum (or a nonlinearly transformed version of it) [Xu et al., 2014]. The original audio phase served to reconstruct a time-domain output signal. Driven by the complex nature of the spectrum and imprecision introduced by a noisy phase, spectral mapping methods adopted real and imaginary part prediction [Fu et al., 2017a, Tan and Wang, 2019].

Although frequency-domain models initially dominated speech enhancement, end-to-end systems directly predicting waveforms later became a point of research interest for their potential benefits stemming from inherently modeling phase information (as phase estimation in the frequency-domain approaches is a non-trivial task). [Fu et al., 2017b] attempted to overcome the phase prediction problem by designing a fully convolutional network for direct waveform enhancement by optimizing MSE between predicted and clean speech signals. Active research continued, building time-domain models on top of different architectures and training paradigms. For instance, SEGAN [Pascual et al., 2017] is a model adopting a generator-discriminator scheme of generative adversarial networks (GAN), the dense convolutional network (DCN) makes use of self-attention modules in an encoder-decoder architecture [Pandey and Wang, 2021], and a U-net-like network is explored in [Défossez et al., 2020].

### 5.1.2 Speaker Separation

Research in speech enhancement revealed that deep learning is a viable approach to advancing the field. While speaker separation has its specific properties (discussed in this section), it shares similarities with speech enhancement. Therefore, trends in both areas were similar and mutually influential. Generally, one architecture is relevant for both tasks (e.g., [Défossez et al., 2020] presented such a study).

The first neural network-based designs employed the T-F representation of the data. The seminal work of [Huang et al., 2014] introduced a feed-forward network and recurrent neural network (RNN) architectures predicting masks of sources given spectral or log Mel-filterbank features. The networks were trained in a signal approximation framework. The limitation of the design was speaker dependence, which is a problem specific to speaker sepa-

[1] Even visually, speech structure is more obvious from plots of real and imaginary components.

ration. Preferably, the system should be capable of separating speakers not seen in training. Moreover, individual speech components in the mixture are equally important; thus, they do not have a specified order. Introducing the deep clustering [Hershey et al., 2016] approach was a milestone, bringing speaker independence into multi-speaker separation with neural networks. Given a log-magnitude spectrum, the deep clustering model predicts an embedding for each T-F bin, where bins dominated by the same speaker should have similar embeddings. During an evaluation, estimated embeddings are clustered by k-means with the number of clusters corresponding to the speaker count. The cluster affiliation subsequently translates to T-F masks per speaker. Although deep clustering overcame the speaker-dependency issue by post-processing T-F embeddings, the speaker permutation problem remained unsolved for approaches associating network outputs with segregated speech segments. A breakthrough solution, permutation invariant training (PIT), was presented in [Yu et al., 2017, Kolbæk et al., 2017]. During PIT, the estimated sources are compared with all the permutations of reference sources. The permutation yielding the lowest loss is selected and used for back-propagation. Although the original work [Yu et al., 2017] used the signal approximation target with real-valued masks, the method is compatible with all mapping and masking approaches. Thus, it has been widely adopted.

Similarly to speech enhancement, frequency-domain models initially dominated speaker separation. Driven by the same reasons, studies employing time-domain signals emerged. TaSNet [Luo and Mesgarani, 2018b] was a pioneering end-to-end model following a signal approximation scheme, replacing STFT and its inverse with a trainable encoder and decoder. Shortly, its fully convolutional successor, Conv-TasNet, was published [Luo and Mesgarani, 2019]. Building upon the success of end-to-end systems, multiple time-domain separators sharing a high-level structure were designed, ruling the speaker separation field: DPRNN utilized a dual-path strategy for modeling long sequences [Luo et al., 2020]. A self-attention mechanism, which became ubiquitous in speech processing, was adopted by DPTNet [Chen et al., 2020] and SepFormer [Subakan et al., 2021]. Despite the trend of time-domain models, TF-GridNet recently resurrected time-frequency models by yielding superb performance [Wang et al., 2023b, Wang et al., 2023a].

## 5.2 Multi-Channel Speech Separation Overview

Various deep-learning approaches, taking multi-channel input and directly predicting a single-channel output, have emerged over the last few years [Gu et al., 2019, Zhang et al., 2020]. While they are suitable for applications where the outputs serve human listeners, using them as pre-processors for downstream models may even lead to performance degradation [Yoshioka et al., 2015]. The reason is that the separated signal is an output of a highly nonlinear function, introducing artifacts. It causes a domain shift of the data w.r.t. the training data of the downstream model. Keeping that in mind, we concentrate on beamforming, which is a viable linear alternative.

As covered in Chapter 3, diverse beamformers optimizing different criteria have been designed over the years. The most commonly used methods in speech processing are broadband versions of the multi-channel Wiener filter (MWF), minimum variance distortionless response (MVDR) [Capon, 1969], and generalized eigenvalue (GEV) [Warsitz and Haeb-Umbach, 2007] beamformers.

As follows from our previous discussion, a precise estimate of the SCMs or steering vector is critical for the beamformer to enhance desired sources. That is where deep learning comes into play. The development of multi-channel neural-based source separation leveraging

beamforming is closely related to the evolution of single-channel models. The early works proposed strong *mask-based beamformers* [Heymann et al., 2016a], building upon the success of frequency domain mask estimators. They gained immediate popularity. They adhere to the following scheme: First, a traditional monaural mask estimator is trained. Various training approaches, including the IBM [Heymann et al., 2016a] and IRM [Erdogan et al., 2016] targets or deep clustering [Higuchi et al., 2017] have been explored. Given real-valued T-F masks indicating the dominance of sources per channel, single source-specific masks are obtained through aggregation. [Heymann et al., 2016a] explored median pooling and [Erdogan et al., 2016] examined max-pooling. Finally, the aggregated values act as weights when estimating source-wise spatial covariance matrices (as will be described in Section 5.4). Given estimated SCMs, beamformer weight computation is straightforward. Following the trend in mask estimation, [Xu et al., 2019] employed complex-valued masks in the beamforming context.

As time-domain separation models became popular, studies trying to leverage their power for the SCM estimation, combining waveform separation and frequency-domain beamforming, have appeared [Ochiai et al., 2020, Mošner et al., 2022a]. Given time-domain source estimates, [Ochiai et al., 2020] proposed two SCM computation approaches based on the STFT images of predictions. The first approach used outputs directly; the second approach computed a ratio of the predicted sources' magnitudes to obtain real-valued masks. The SCM computation then followed the traditional mask-based beamforming scheme.

With the renaissance of direct spectral mapping approaches, using their outputs for SCM estimation became appealing [Wang et al., 2021c]. It represents a more straightforward alternative than using masks or time-domain predictions since the complex-domain outputs are directly usable to estimate the SCMs.

The abovementioned systems leverage source separation to enable SCM computation while relying on a specific beamformer formulation. This reliance poses an upper bound on performance. To overcome it, multiple studies took a beamforming approach with the weights estimated by a neural network [Luo et al., 2019, Luo et al., 2020, Sainath et al., 2017]. Following the same line of research, *all-neural beamforming* integrating trainable beamformers into a neural network has recently been introduced [Zhang et al., 2021b, Zhang et al., 2021a, Xu et al., 2021, Gu et al., 2023]. As a promising concept, it gradually gains more attention from the community.

## 5.3 Evaluation Metrics

In general, source separation techniques are commonly assessed utilizing:

- subjective evaluation,
- objective evaluation.

While the objective evaluation aims to assess enhanced speech via pre-defined deterministic calculation, subjective metrics provide human opinions. Human evaluation is considered a "golden standard" and requires no reference clean recording for comparison. On the other hand, the prohibitive expense of obtaining reliable outcomes and poor scalability make researchers resort to well-defined, easy-to-obtain objective metrics. For the same reasons and also due to focus on the SV task, we will consider only the objective evaluation in our experiments. Nevertheless, for completeness, we will briefly present both types of evaluation.

### 5.3.1 Subjective Evaluation

Subjective evaluation of enhancement/separation techniques is especially valuable in human perception applications (as human beings are consumers of the output audio). Nevertheless, human evaluation is also employed in speech separation challenges as a means of comparison [Leglaive et al., 2025]. From the application perspective, it is worth noting that high subjective scores do not guarantee improvements in downstream models performance when enhancement serves as pre-processing.

In order to make human evaluation interpretable and principled, the Telecommunication Standardization Sector of the International Telecommunication Union (ITU-T) issued multiple recommendations. Recommendation P.800[2] [International Telecommunication Union, 1996] provided guidelines for subjective evaluation of transmission quality. In the testing phase, human evaluators are asked to assess the quality on a five-point scale, assigning one of the following ratings to an audio clip: excellent (5), good (4), fair (3), poor (2), and bad (1). The averaged quantity is referred to as the *mean opinion score* (MOS). Reflecting on the scalability issue and availability of various platforms, recommendation P.808[3] [International Telecommunication Union, 2021] described a crowdsourcing approach to the subjective evaluation of speech quality while adopting the opinion scales of P.800. On top of extending P.800 to crowdsourcing, it also complements another recommendation we will cover, P.835 [International Telecommunication Union, 2003].

Assigning one categorical score to a recording does not provide information about the aspects that contributed the most to such a decision. When human judgment is used to assess speech enhancement, subjects might be unsure what to evaluate, as there is a common trend that speech degradation increases with more noise that is suppressed [International Telecommunication Union, 2003]. To tackle this issue, ITU-T issued recommendation P.835[4], which disentangles the evaluation of speech enhancement to individual assessments of *speech signal*, *background noise*, and *the overall effect*. Similar to the MOS of P.800, each of the three evaluations employs a five-point discrete scale.

### 5.3.2 Objective Evaluation

Objective metrics can be divided into *intrusive* and *non-intrusive*. The former requires a clean reference, whereas the latter estimates a specific score using just the enhanced/separated audio clip.

#### Intrusive Metrics

The most common objective metrics used in research papers are the intrusive ones. They have multiple benefits:

- They are widely adopted, making comparison and interpretation of various approaches easy.
- Some metrics are standardized, which makes them well-defined.
- The availability of accessible implementations makes adoption smooth.

[2] Methods for subjective determination of transmission quality

[3] Subjective evaluation of speech quality with a crowdsourcing approach

[4] Subjective test methodology for evaluating speech communication systems that include noise suppression algorithm

- They are usually not overly computationally expensive and do not require any training stage before being used.

The intrusive metrics inherently also have drawbacks. In order to obtain the score, a clean reference must be available. It is usually available only for simulated data, limiting the informative value of evaluation as real-world (out of domain) recordings (without the reference) tend to be the most difficult [Zhang et al., 2024]. A limited correlation between intrusive metrics and human evaluation of speech quality has also been reported in [Cano et al., 2016]. Such results are alerting for perceptual applications. However, separated/enhanced outputs do not always directly serve human listeners.

We will use intrusive metrics to assess separation performance for the outlined benefits. Another reason is that separated signals are only intermediate representations in our pipeline and not the final outputs. One of the most common metrics is the signal-to-distortion ratio (SDR) [Vincent et al., 2006], which will be discussed later. The version of a ratio metric considering possibly different dynamic ranges of reference and prediction is the scale-invariant signal-to-distortion ratio (SI-SDR) [Roux et al., 2019], sometimes also called the scale-invariant signal-to-noise ratio (SI-SNR). ITU-T standardized multiple metrics to estimate the perceptual quality of speech. Perceptual Evaluation of Speech Quality (PESQ) [Rix et al., 2001], recommendation P.862 [International Telecommunication Union, 2001], and its extension to wideband data (wideband PESQ, recommendation P.862.2) [International Telecommunication Union, 2007] are representatives of widely adopted metrics in the community. An update to PESQ considering super-wideband audio is termed Perceptual Objective Listening Quality Assessment (POLQA), issued as recommendation P.863 [International Telecommunication Union, 2018]. A metric predicting intelligibility is called Short-Time Objective Intelligibility (STOI) [Taal et al., 2010]. Since we will use SDR and STOI to assess our separation techniques, we will describe only these two in detail.

**Signal-to-Distortion Ratio (SDR)**

As established in [Vincent et al., 2006], the signal-to-distortion ratio is a performance measure using an energy ratio between the target signal and a residual signal. The proposed criterion allows for four acceptable distortion types: time-invariant/-variant gains and time-invariant/-variant filters. To this end, the enhanced/separated signal is decomposed into the target signal (a version of a dry signal with allowed distortions) and error terms representing interferences, noise, and artifacts. It is up to the user to decide what kind of distortions are allowed. Studies in the source separation field commonly adopt the implementation of the `bss_eval`[5] toolbox, choosing time-invariant filtering distortions. Given the time-domain reference[6] $\mathbf{x} \in \mathbb{R}^{\mathsf{T}}$ and its prediction $\hat{\mathbf{x}} \in \mathbb{R}^{\mathsf{T}}$, the SDR is computed as:

$$\mathrm{SDR}(\mathbf{x}, \hat{\mathbf{x}}) = 10 \log_{10} \frac{\|\mathbf{h}_{\mathrm{SDR}} \star \mathbf{x}\|^2}{\|\hat{\mathbf{x}} - \mathbf{h}_{\mathrm{SDR}} \star \mathbf{x}\|^2}, \tag{5.5}$$

where the symbol $\star$ represents convolution and $\mathbf{h}_{\mathrm{SDR}}$ is a 512-tap FIR filter that maximizes the metric. It is a common practice to report the improvement in SDR over the mixture SDR:

$$\mathrm{SDRi}(\mathbf{x}, \hat{\mathbf{x}}, \mathbf{y}) = \mathrm{SDR}(\mathbf{x}, \hat{\mathbf{x}}) - \mathrm{SDR}(\mathbf{x}, \mathbf{y}). \tag{5.6}$$

[5] https://gitlab.inria.fr/bass-db/bss_eval

[6] Note that $\mathbf{x}$ is a time-domain version of $\mathbf{X}$ in our signal model (Section 3.3.1) where we omit the channel index.

We will employ this criterion when reporting source separation results. We will usually use the term *mean SDRi* to stress that the final metric was obtained by averaging per-source SDRi values.

**Short-Time Objective Intelligibility (STOI)**

As the name implies, the Short-Time Objective Intelligibility (STOI) metric [Taal et al., 2010, Taal et al., 2011] is used to objectively assess speech intelligibility. It compares a processed speech signal with a clean reference by analyzing corresponding short-time spectral representations. It was designed for a sampling rate of 10 kHz so that the represented frequencies correspond to those relevant for speech intelligibility.

The metric is computed according to the following steps:

1. STOI operates by first decomposing the target $\mathbf{x} \in \mathbb{R}^{\mathsf{T}}$ and predicted $\hat{\mathbf{x}} \in \mathbb{R}^{\mathsf{T}}$ speech signals into short-time half-overlapping frames (of 256 samples padded to 512 samples) and transforming them with DFT to $\mathbf{X} \in \mathbb{R}^{T \times F}$ and $\hat{\mathbf{X}} \in \mathbb{R}^{T \times F}$.

2. The spectra $\mathbf{X}$ and $\hat{\mathbf{X}}$ are mapped onto a third-octave frequency scale, which simulates human auditory perception:

$$X_{t,j} = \sqrt{\sum_{f=f_j^{(\text{low})}}^{f_j^{(\text{high})}} |X_{t,f}|^2}, \qquad \hat{X}_{t,j} = \sqrt{\sum_{f=f_j^{(\text{low})}}^{f_j^{(\text{high})}} |\hat{X}_{t,f}|^2}, \tag{5.7}$$

   where $f_j^{(\text{low})}$ and $f_j^{(\text{high})}$ denote edge frequency bins for the $j$th third-octave band.

3. Given $X_{t,j}$ and $\hat{X}_{t,j}$ with grouped frequency bins, intermediate intelligibility measures are computed. To this end, the time context is considered for each band. The context is used to normalize the energy of the predicted signal to match that of clean speech in each band. Subsequently, clipping is applied to lower-limit the signal-to-distortion ratio (see [Taal et al., 2010] for details). The two steps yield $\hat{X}'_{t,j}$ for each $t$ and $j$. The intermediate intelligibility measures are then computed as

$$d_{t,j} = \frac{\sum_\tau \left[ (X_{\tau,j} - \bar{X}_{t,j})(\hat{X}'_{\tau,j} - \bar{\hat{X}}'_{t,j}) \right]}{\sqrt{\sum_\tau (X_{\tau,j} - \bar{X}_{t,j})^2 \sum_\tau (\hat{X}'_{\tau,j} - \bar{\hat{X}}'_{t,j})^2}}, \tag{5.8}$$

   with $\tau$ representing frame indices within the context window. $\bar{X}_{t,j}$ and $\bar{\hat{X}}'_{t,j}$ denote the mean values of $X_{\tau,j}$ and $\hat{X}'_{\tau,j}$ over the context window, respectively.

4. Finally, a scalar metric value is a mere average of intermediate measures from the previous step over time and over frequency:

$$\text{STOI}(\mathbf{x}, \hat{\mathbf{x}}) = \frac{1}{TJ} \sum_{t,j} d_{t,j}, \tag{5.9}$$

   where $T$ and $J$ denote the total number of frames and third-octave bands, respectively. Stemming from (5.8), the averaged value of STOI takes values between 0 and 1, with 1 representing perfect intelligibility.

STOI is widely used in speech processing due to its robustness, efficiency, and correlation with human intelligibility scores. However, it does not directly measure overall audio quality or distortions unrelated to intelligibility. Variants like eSTOI (Extended STOI) [Jensen and Taal, 2016] improve performance in low-SNR conditions.

#### Non-Intrusive Metrics

Non-intrusive metrics represent a middle ground between subjective and intrusive objective metrics. They are handy in real speech applications since they do not require a clean speech reference. The non-intrusive metrics are usually provided by models pre-trained to predict the underlying quality measure before being deployed as a means for evaluation.

We can identify two groups of non-intrusive metrics: 1) those that approximate human judgment and 2) those predicting intrusive metrics without needing a clean reference. First, we will mention a few examples of the former group. [Reddy et al., 2021] trained a CNN-based MOS predictor on highly varied data processed by over 200 noise suppressors and rated with MOS values by human evaluators following recommendation P.808. The resulting network is hence named DNSMOS. Reacting to the extended recommendation P.835, the same authors also trained an updated version of DNSMOS. Therefore, the resulting model, termed DNSMOS P.835 [Reddy et al., 2022], predicts three scores: speech quality, background noise, and overall quality.

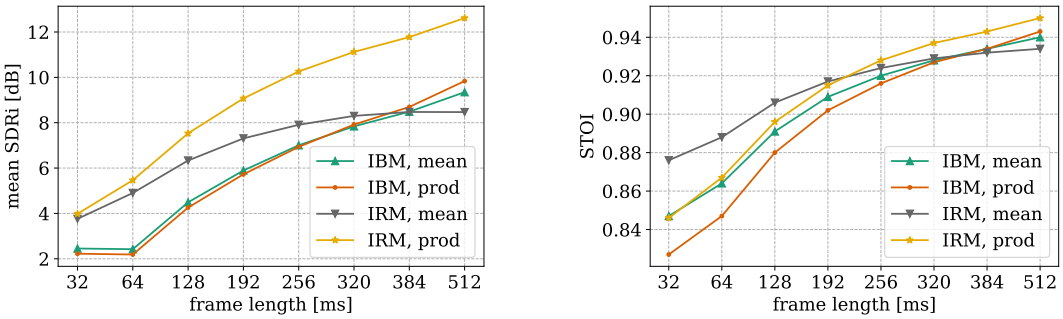
Falling within the second group, Quality-Net [Fu et al., 2018] represents an attempt to approximate the wideband PESQ metric (recommendation P.862.2) via training a bi-directional LSTM network on data annotated with the corresponding quality metric. [Kumar et al., 2023] developed a suite of models predicting standard metrics in quality/intelligibility assessment and integrated it into the widely adopted TorchAudio library as TorchAudio-Squim[7]. Out of the objective metrics, it provides estimates of PESQ, STOI, and SI-SDR. Under the hood, there is one neural network model with multiple outputs that was trained in a multi-task fashion. Three outputs correspond to the three metrics of interest. The last output corresponds to the prediction of the reference signal. Predicting it along with metrics helped to regularize the training of the shared backbone.

## 5.4 Oracle Beamforming and Hyperparameters

Although the MVDR is well-established, it operates on the outputs of STFT in the real-world broadband setting (see Section 3.3). Therefore, the STFT hyperparameters must be determined first. In contrast to speech processing applications, such as ASR, frame length and shift settings for beamforming are not unified in the literature. For instance, [Higuchi et al., 2017, Wang et al., 2021c] used windows of 32 ms and shifts of 8 ms, [Heymann et al., 2016a] resorted to 64 and 16 ms, and [Ochiai et al., 2020] scaled the size and shift even to 512 ms and 128 ms. We also hypothesize that the setting is task- and data-specific (e.g., due to required system delay and the environment stationarity).

Therefore, before experimenting with training models for mask estimation in the context of beamforming, we analyzed the impact of window setting on separation performance using the oracle MVDR beamformer. A premise of the oracle beamformer is that sources used to compute masks according to (5.1) and (5.2) for IBM and IRM, respectively, are known exactly. We take advantage of having speech and noise references (before mixing) for the simulated part of the MultiSV (*eval simu*). Specifically, we use only the test sides of trials. Given $N$ computed ideal masks for speech $\{\mathbf{M}_n^{(X)}\}_{n=1}^N$ and $N$ masks for noise $\{\mathbf{M}_n^{(V)}\}_{n=1}^N$, the pooling operation must be applied to obtain global masks for both types of sources. We experimented with two pooling types: mean $\mathbf{M}^{(\nu)} = \frac{1}{N}\sum_n \mathbf{M}_n^{(\nu)}$, and product pooling $\mathbf{M}^{(\nu)} = \prod_n \mathbf{M}_n^{(\nu)}$ (where we used symbol $\nu$ to substitute speech $X$ or noise $V$). Finally,

[7] https://pytorch.org/audio/main/tutorials/squim_tutorial.html




(a) Mean SDRi

(b) STOI

Figure 5.2: Separation metrics for the oracle MVDR beamformer as a function of the frame length used in STFT. Different curves correspond to different combinations of masking and pooling approaches (*prod* stands for product pooling). The frame step is always set to a quarter of the frame length, which is widely accepted in the literature.

values of the combined masks $\mathbf{M}^{(\nu)}$ are used as weights in the SCMs estimation. Following our signal model, $\mathbf{y}_{t,f} \in \mathbb{C}^N$ represents an $N$-dimensional vector grouping a specific T-F bin values of all channels, see (3.61). $M_{t,f}^{(\nu)} \in \mathbb{R}$ represents a mask value corresponding to one T-F bin. Then, both frequency-dependent speech and noise SCMs are obtained as

$$\mathbf{\Phi}_f^{(\nu\nu)} = \frac{\sum_{t=1}^{T} M_{t,f}^{(\nu)} \mathbf{y}_{t,f} \mathbf{y}_{t,f}^{\mathsf{H}}}{\sum_{t=1}^{T} M_{t,f}^{(\nu)}}, \tag{5.10}$$

with $(\cdot)^H$ representing the conjugate (or Hermitian) transpose.

The outcomes of the experiments with the oracle beamforming are presented in Figure 5.2. First, we notice a rising trend for all combinations and both separation metrics. Hence, separation performance increases with increased frame length (in other words, with increased frequency resolution and decreased time resolution). We note that analysis windows are considerably larger than those used in other speech-processing tasks, such as ASR or speech separation. However, speech stationarity within a frame is not a concern. In contrast, the prevalence of speech vs noise per T-F bin is important. Arguably, better frequency resolution provides a clearer view of speech (e.g., fundamental and harmonic frequencies) and noise structure, resulting in fine-grained information about what source dominates at which frequency.

The mean SDRi results in Figure 5.2a suggest that the pooling type is not a critical hyperparameter when employing IBM masks. Despite this, the product pooling becomes slightly better than the mean pooling as the frame length increases. The same trend is observable for the STOI metric (Figure 5.2b). However, the difference between the two pooling types becomes more prominent. In contrast, focusing on the IRM masks, a different impact of the pooling type is evident. Although mean pooling provides good results for small window sizes (especially STOI), the performance curves in Figure 5.2 quickly plateau. On the other hand, product pooling yields steadily increasing performance and clearly outperforms other approaches in terms of mean SDRi.

Our analysis demonstrated that IRM masks combined with product pooling have the potential to provide the best results. However, we should stress that the experiments assumed ideal conditions — simulated MultiSV data and perfect estimation of sources.

Therefore, we will revalidate the outcomes in Section 5.6.1, where we will estimate SCMs with the help of neural networks.

## 5.5 Separation Models

Motivated by the discussed advantages of spatial filtering, we focused on building beamforming-based pre-processing for subsequent speaker embedding extraction. We followed the evolution in multi-channel source separation, where we aimed to employ neural networks in the crucial SCM estimation process. Although our goal of separating speech from noise differs from most works we followed (targeting speech-speech separation), the underlying principles are similar. Therefore, we explored mask-based beamforming (see Section 5.6.1, taking inspiration from [Heymann et al., 2016a, Erdogan et al., 2016]) and then the combination of time-domain separation and frequency-domain beamforming (see Sections 5.6.2 and 5.6.3 inspired by [Ochiai et al., 2020]). Eventually, we proposed extensions to the latter (Section 5.7 and [Mošner et al., 2023]). Before diving into principles, our original work, and motivations for our design choices, we will briefly describe the core neural networks we utilized in the speech and noise SCM prediction process.

### 5.5.1 Recurrent Mask Estimator

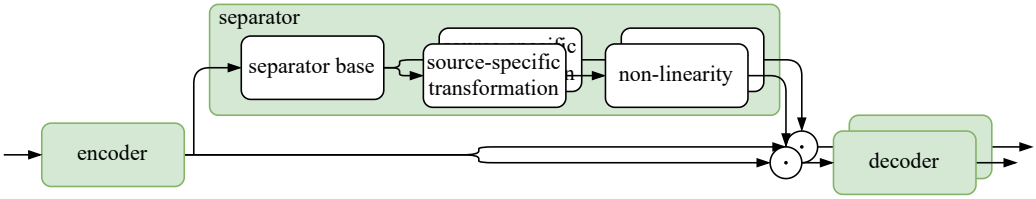
Although one can think of a plethora of architectures for speech and noise mask estimation, we took inspiration from [Heymann et al., 2016a]. The study employed recurrent neural networks since they were mainstream models for sequence modeling at the time of publication and of designing our technique.

The simple neural mask estimator operates in the frequency domain. The input to the network is a mean- and variance-normalized magnitude spectrum of a mixture. First, the input is processed by one recurrent layer, long short-term memory (LSTM) [Hochreiter and Schmidhuber, 1997] with a hidden dimensionality of 513 in our case. The output is then passed to a sequence of two affine layers. Each one projects the input to a 513-dimensional space. Each projection is followed by a rectified linear unit (ReLU) nonlinearity [Fukushima, 1969, Nair and Hinton, 2010]. The final part consists of two parallel affine layers with sigmoid activation functions that provide speech and noise mask predictions. We note that the input and output dimensionalities are equal. We varied them in our experiments; therefore, the specific input/output sizes will be provided later in the text.

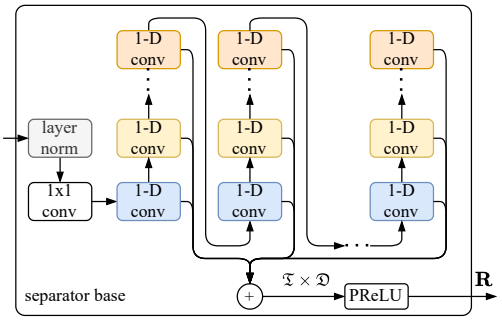
We explored two ways of training following the target mask definition in Section 5.1.1. 1) The classification approach to training the network uses IBM references and the binary cross-entropy (BCE) loss function. 2) A regression-style training takes IRMs as targets and minimizes mean squared error (MSE). In the text, we will explicitly differentiate between the two methods.

### 5.5.2 Time-Domain Separation Models

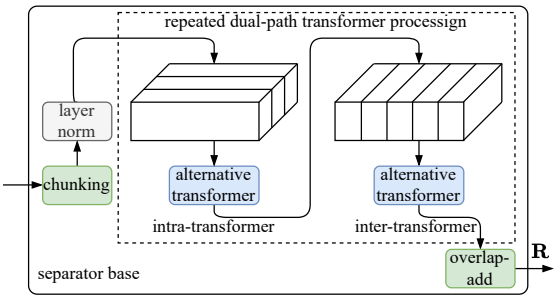
In general, time-domain separation models share the same high-level structure shown in Figure 5.3a. The initial part, *encoder*, is responsible for an operation resembling spectral analysis. It can be learned or employ the Fourier bases. [Heitkaemper et al., 2020] analyzed the effect of the encoder/decoder choices and concluded that the type of latent space is not the most crucial aspect determining separation performance. A trainable encoder is usually implemented by one strided convolutional layer. Both the bare linear version [Luo




(a) Common high-level structure of time-domain separation models.




(b) Separator base module of Conv-TasNet. Adapted from [Luo and Mesgarani, 2019].




(c) Separator base module of DPTNet. Adapted from [Chen et al., 2020].

Figure 5.3: Scheme of blocks identifiable in time-domain separation models. We explicitly detail separator base modules for the two architectures we employed.

and Mesgarani, 2019] and alternatives applying non-linearity to the encoded input [Luo and Mesgarani, 2018b, Luo and Mesgarani, 2018a] have been investigated. Interestingly, empirical evidence suggests that a tiny analysis window (as small as 0.25 ms) is preferable [Heitkaemper et al., 2020, Chen et al., 2020].

The second part is a *separator*. It is the main module that differentiates variants of the time-domain models. For convenience, we divide it into three parts (as shown in Figure 5.3a):

**Separator base** The input is immediately normalized, for instance, by layer normalization [Ba et al., 2016] or global layer normalization [Luo and Mesgarani, 2019]. The normalization is followed by a network based on a temporal convolutional network (TCN) [Luo and Mesgarani, 2019], LSTM layers [Luo and Mesgarani, 2018b, Luo et al., 2020], or Transformer encoder layers [Chen et al., 2020, Subakan et al., 2021], to name a few. The examples of architectures with TCN and Transformer encoders are shown in Figures 5.3b and 5.3c, respectively. We refer to the output as $\mathbf{R} \in \mathbb{R}^{\mathfrak{T} \times \mathfrak{D}}$, where $\mathfrak{T}$ represents the number of frames (as extracted by the encoder), and $\mathfrak{D}$ is the dimensionality of the internal features.

**Source-specific transformation** The representation $\mathbf{R}$ provides summary information about sources that should be easily decoupled. It is commonly achieved by linear (or affine) transformation [Luo and Mesgarani, 2019, Chen et al., 2020].

**Non-linearity** Finally, in order to obtain masks for each source given real-valued representations, a non-linear operation $\varsigma(\cdot; \theta)$ is applied. It can be as simple as a nonlinear function (sigmoid, ReLU, or softmax) or use a gating mechanism parametrized by $\theta$ [Chen et al., 2020].

Table 5.1: Hyperparameters of the used time-domain separation models.

| | | DPTNet low | DPTNet | | Conv-TasNet | Conv-TasNet small |
|---|---|---|---|---|---|---|
| codec | bases | 64 | 64 | bases | 512 | 256 |
| | frame | 32 | 8 | frame | 32 | 40 |
| internal | feature dim | 64 | 64 | B | 128 | 128 |
| | hidden dim | 256 | 128 | Sc | 128 | — |
| | segment size | 250 | 250 | H | 512 | 192 |
| | heads | 4 | 4 | P | 3 | 3 |
| size | layers | 2 | 6 | X | 8 | 7 |
| | | | | R | 3 | 3 |
| parameters [M] | | 2.85 | 2.79 | | 5.00 | 1.20 |

The masks predicted by the separator are applied to the encoder output by multiplication, providing separated sources in a latent space.

In order to obtain time-domain signals, the internal representations are transformed by a *decoder*. Although it should be symmetrical to the encoder, it is not necessary for it to be the inverse. Luo showed that making both the encoder and the decoder trainable is superior to constraining the decoder to perform the pseudo-inverse operation of the encoder [Luo and Mesgarani, 2019].

### Conv-TasNet

We chose Conv-TasNet [Luo and Mesgarani, 2019] as the first time-domain model for our experiments, since it represented a significant milestone in source separation. Not only did it outperform previous T-F masking approaches and several ideal (magnitude) masks, but it is also a compact model with low latency.

The encoder and decoder blocks are implemented by trainable convolutional and transposed convolutional layers, respectively. A notable property of the separator base module is that it utilizes TCN [Lea et al., 2017] (as opposed to previously used recurrent layers) to capture long-range context. Figure 5.3b shows that the network comprises multiple stacks of *1-D conv* blocks. The blocks in each such stack incorporate a 1D dilated convolution with dilation increasing by a factor of two, gradually enlarging the receptive field. Each 1-D conv block starts by transforming the input with a 1×1 convolution. Then, instead of a standard 1D convolution, the authors used a parameter-efficient alternative, depthwise separable convolution (with dilation) [Sifre, 2014]. The main idea is to decouple a 1D convolution into two operations. First, input channels are independently convolved with channel-specific kernels of arbitrary size (i.e., depthwise convolution). Then, in order to combine channels, a 1×1 convolution is applied. In Conv-TasNet, the two operations are interleaved with PReLU non-linearity and normalization. It is worth noting that each 1-D conv block comprises two final 1×1 convolutional layers. The output of the first one is summed with the block input through a residual connection and passed to the following block. The second one provides intermediate output. All such outputs are eventually summed to yield the output of the TCN. PReLU nonlinear function then converts it to the internal representation $\mathbf{R}$. Source-specific transformation is implemented by linear projection into pre-mask matrices. Finally, the sigmoid function is applied to obtain mask values.

In Table 5.1, we provide hyperparameters of the network versions we employed in our experiments, where we follow the naming convention defined in [Luo and Mesgarani, 2019]. Notably, we increased frame length from 16 to 32 samples due to a sampling rate of 16 kHz that is twice that of the original Conv-TasNet.

In speaker separation, time-domain models, including Conv-TasNet, are commonly trained to optimize the SI-SNR objective [Roux et al., 2019] in the PIT scheme. However, since our training-targets $\mathbf{x} \in \mathbb{R}^{\mathsf{T}}$ and $\mathbf{v} \in \mathbb{R}^{\mathsf{T}}$ are well-defined, we employ an alternative ratio objective:

$$\mathcal{L}_{\text{SEP}}(\mathbf{x}, \mathbf{v}, \hat{\mathbf{x}}, \hat{\mathbf{v}}) = -\frac{1}{2}\Big(10 \log \frac{\|\mathbf{x}\|_2^2}{\|\hat{\mathbf{x}} - \mathbf{x}\|_2^2} + 10 \log \frac{\|\mathbf{v}\|_2^2}{\|\hat{\mathbf{v}} - \mathbf{v}\|_2^2}\Big), \tag{5.11}$$

where $\hat{\mathbf{x}}$ and $\hat{\mathbf{v}}$ are time-domain predictions of reverberant speech and noise, respectively.

### Dual-Path Transformer Network

As the Transformer architecture became adopted in many applications requiring sequence modeling, attempts to utilize it in source separation also emerged. The problem with time-domain source separation is that the length of sequences that need to be processed can be in the tens of thousands. It is because of small window sizes (as mentioned before) and sampling frequency (e.g., 8 kHz or 16 kHz [Hershey et al., 2016, Cosentino et al., 2020]). Direct use of Transformers becomes troublesome for modeling such long sequences due to the quadratic growth of memory. Dual-Path Transformer Network (DPTNet) [Chen et al., 2020] managed to employ Transformers by leveraging a dual-path scheme to sequence processing (shown in Figure 5.3c) presented earlier in the source separation field [Luo et al., 2020]. The main idea is to split the sequence into chunks and process them in two steps: 1) Intra-chunk operation processes chunks independently, where short-range context can be considered. 2) Inter-chunk operation then processes the data across chunks. This allows for contextualized representations with long-range dependencies. In contrast to DPRNN [Luo et al., 2020], DPTNet uses an alternative version of the Transformer encoder block to perform intra- and inter-chunk processing. In the original Transformer encoder block, a series of two affine layers with non-linearity in between transform each token after multi-headed attention. Since the bare Transformer encoder has no notion of order, inputs are usually augmented with positional embeddings. In DPTNet, the authors found it difficult to train the model employing positional embeddings [Chen et al., 2020]. Therefore, they decided to modify the abovementioned sub-network of two linear layers by replacing the first one with a recurrent neural network (RNN). It inherently provides order-aware processing since RNNs model sequences by design.

In summary, the separator base, shown in Figure 5.3c, comprises chunking that splits the input sequence to allow dual-path processing, followed by layer normalization. Subsequently, a tandem of intra-transformer and inter-transformer is repeated multiple times to perform separation. Then, in the original study [Chen et al., 2020], an affine transformation provides source-specific representations passed to an overlap-add operation. We note that the inverted order of the last two operations provides equivalent output when properly rescaling the bias. Therefore, the DPTNet architecture also conforms to the general structure of time-domain separators outlined before when integrating the overlap-add operation into the separator base. Eventually, DPTNet uses a nonlinearity $\varsigma(\cdot; \theta)$ parametrized by $\theta$. Specifically, a gating mechanism (comprising parallel branches, each linearly transforming

data and applying either tanh or sigmoid functions, whose outputs are multiplied) followed by the ReLU activation function provides source masks.

The encoder and decoder are similar to those of Conv-TasNet, but they operate with extremely short frames. The original DPTNet used a frame length of 2 samples when processing the audio sampled at 8 kHz. This translates to 4 samples in our case of 16 kHz sampled data. Since we observed no improvements in the SV results over the same model utilizing a frame size of 8 samples, we resorted to longer frames, decreasing memory requirements. As the DPTNet model is still demanding regarding memory and computation, we also designed a modified version, DPTNet low, for faster experimentation. Its hyperparameters and those for our DPTNet are presented in Table 5.1. All DPTNet models were trained to optimize the objective in (5.11).

## 5.6 Using Outputs of Separation Models for Beamforming

In this section, we are interested in combining the capacity of neural network-based speech-noise separators and linear spatial filtering, i.e., beamforming. We will stick with one theoretically grounded broadband beamformer, MVDR, as defined in (3.63), to allow focusing on the preceding computation steps that we aim to analyze. According to the definition, spatial covariance matrices (SCMs) of speech and noise are unknown; hence, they need to be estimated. As follows from the MVDR formula, beamforming performance is directly affected by the quality of SCM estimates. Therefore, we will explore various ways to use neural networks to predict SCM. Based on the type of source separator, we will differentiate between three approaches:

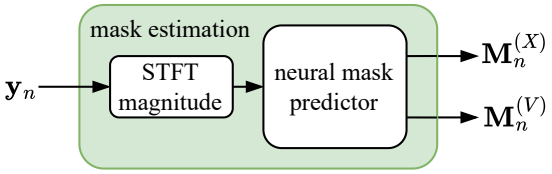
- Direct mask prediction for mask-based beamforming (Figure 5.4a)
- Indirect mask prediction for mask-based beamforming (Figure 5.4b)
- Mask-less beamforming

The former uses mask-predicting networks as described in Section 5.5.1. The other two, reflecting the development in the source separation field, use time-domain models (from Section 5.5.2).

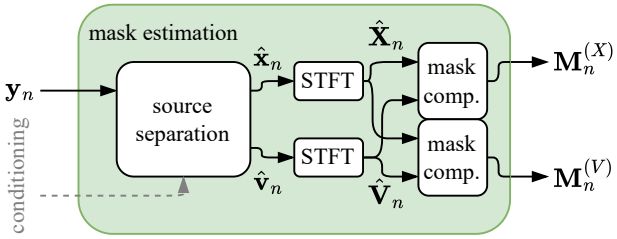
In all the following experiments, we employed MultiSV (see Section 4.3.4) data to train the separation models, employing respective loss functions described in Sections 5.5.1 and 5.5.2. In the case of single-channel models, we randomly selected one channel from the microphone array per example. For multi-channel models, we used all channels as inputs and predicted the sources' signals at the reference microphone.

### 5.6.1 Direct Mask Prediction for Mask-Based Beamforming

Direct mask estimation, as depicted in Figure 5.4a, is inspired by studies from the field of ASR [Heymann et al., 2016a, Erdogan et al., 2016]. The approaches presented in this section employ direct mask predictors. The input to the predictor is a mean and variance normalized mixture magnitude spectrum. Given such T-F representation, the network predicts one mask $\mathbf{M}_n^{(X)} \in \mathbb{R}^{T\times F}$, which signifies speech activity, and one mask $\mathbf{M}_n^{(V)} \in \mathbb{R}^{T\times F}$ corresponding to noise activity for the $n$th microphone. We note that one mask does not necessarily complement the other mask since some T-F bins might not be dominated by either of the sources (e.g., during intervals of silence). Therefore, we use two network outputs.




(a) Direct estimation.




(b) Indirect estimation.

Figure 5.4: Explored ways of speech and noise mask estimation. We note that the network weights are shared across channels. The conditioning branch in (b) is used only when employing reference channel attention (see Section 5.7.1 for details).

We process channels of the microphone array independently, yielding $N$ masks for speech $\{\mathbf{M}_n^{(X)}\}_{n=1}^N$ and $N$ masks for noise $\{\mathbf{M}_n^{(V)}\}_{n=1}^N$. Subsequently, we pool both types of masks using the same pooling operation to provide global masks. Eventually, the global masks are used according to (5.10) to provide estimates of SCMs for speech and noise.

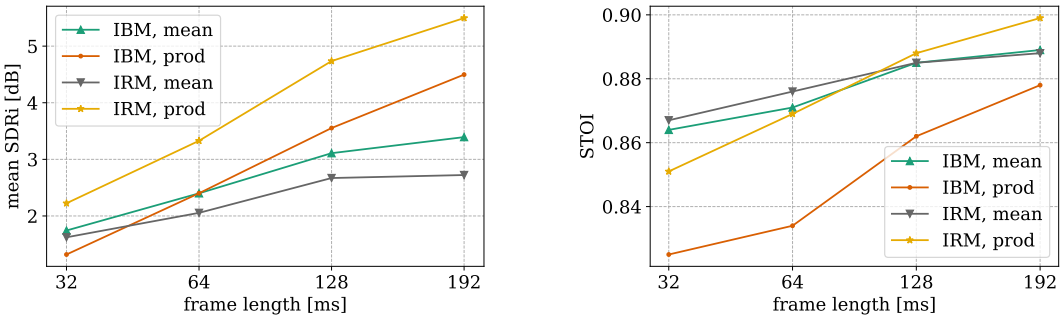
Notably, the strategy of pooling masks and then weighting outer products is not the only way to estimate SCMs given per-channel masks. One may also consider adopting the approach employed by single-channel separation/enhancement methods with real masks: SCMs might be obtained directly using estimates of speech/noise provided by masking the mixture magnitude spectrograms and using the mixture phase. Such an SCM prediction (without pooling operation) is then given as

$$\mathbf{\Phi}_f^{(\nu\nu)} = \frac{1}{T}\sum_{t=1}^{T}(\mathbf{m}_{t,f}^{(\nu)} \odot \mathbf{y}_{t,f})(\mathbf{m}_{t,f}^{(\nu)} \odot \mathbf{y}_{t,f})^{\mathsf{H}}, \tag{5.12}$$

where $\mathbf{m}_{t,f}^{(\nu)} \in \mathbb{R}^N$ is constructed analogically to $\mathbf{y}_{t,f}$ from per-channel masks $\mathbf{M}_n^{(\nu)}$. The operator $\odot$ performs element-wise multiplication. We empirically found that this approach consistently underperforms the method in (5.10). Therefore, we mention it for completeness and do not consider it further. Instead, we focus on the former way of SCM estimation.

### Models for Mask Estimation

The architecture of the employed mask estimator follows the description in Section 5.5.1. In Section 5.4, we showed that the frame length and step noticeably affect the performance of the beamformer in an oracle setting (measured by mean SDRi and STOI), which makes these hyperparameters worth exploring in more general scenarios. To allow for such experiments, we implemented and trained various versions of the base model that differ in the dimensionality of input and output layers (which alters the number of parameters as shown in Table 5.2), reflecting specific frame lengths.




(a) Mean SDRi

(b) STOI

Figure 5.5: Separation metrics for the MVDR beamformer as a function of the frame length used in STFT and by mask estimators. Different curves correspond to different combinations of training targets (IBM or IRM masks) and pooling approaches (*prod* stands for product pooling). The frame step is always set to a quarter of the frame length, which is widely accepted in literature.

Table 5.2: The impact of the frame size on the number of parameters of the direct mask predictor.

| frame length | 32 ms | 64 ms | 128 ms | 192 ms |
|---|---|---|---|---|
| number of parameters [M] | 2.37 | 3.16 | 4.74 | 6.32 |

In order to allow comparison inspired by experiments with the oracle binary and ratio masks in Section 5.5.1, we took two approaches to training the mask estimator. 1) We followed the classification training style. Hence, binary cross-entropy between outputs and classes provided by the IBM served as a loss function. 2) The second approach optimized the mean squared error between network outputs and IRMs.

### Experiments

#### Frame Length Dependence

First, our objective was to answer the question of how indicative the oracle experiments are about the performance when employing machine learning to estimate SCMs. Therefore, we trained multiple models described above. We used their per-channel predictions as masks that we pooled, and performed SCM estimation following (5.10). For the analysis, we employed the MVDR beamformer in (3.63).

The mean SDRi and STOI results are presented in Figures 5.5a and 5.5b, respectively. First, we notice that the performance is considerably worse than that in the oracle scenario in Figure 5.2. Nevertheless, it is reasonable and expected. Mean SDRi results reveal trends comparable to those of the oracle ones, except for the combination of IBM targets and mean pooling. Specifically, the performances of IRM and IBM with the product pooling tend to increase at a similar pace with the increase of frame length, with IRM consistently outperforming IBM. Also, IRM with mean pooling tends to plateau early. Similar conclusions can be drawn for the STOI curves. Overall, despite the variability introduced by the optimization of networks and the counts of parameters, the results confirmed outcomes stemming from the oracle experiments about the superiority of the IRM and product pooling pair.

Table 5.3: Comparison of direct (single-channel) speech separation using predicted masks to reconstruct time-domain signals and beamforming with different approaches to mask pooling. The frame length for beamforming is 128 ms, and the frame shift is 32 ms.

| | **IBM** | | | **IRM** | | |
|---|---|---|---|---|---|---|
| method | beamforming | | direct | beamforming | | direct |
| pooling | mean | product | ✗ | mean | product | ✗ |
| mean SDRi [dB] | 3.11 | 3.55 | **6.32** | 2.67 | 4.74 | **5.80** |
| STOI | **0.885** | 0.862 | 0.808 | 0.885 | **0.888** | 0.872 |

We will use ratio masks combined with product pooling in the following experiments based on our analyses here and in Section 5.4. Although they do not provide the best results, we opt for a frame length of 128 ms and a shift of 32 ms. This setting already provides a reasonable performance boost compared to a window of 32 ms. At the same time, we believe it is more practical than longer windows. Although the scenes in our data are static, changes in acoustics can occur quickly in real-world scenarios, which calls for shorter window sizes.

**Single-Channel vs. Multi-Channel Separation**
Since we trained masked estimators on single-channel data, it is possible to use masks produced by the networks according to single-channel separation/enhancement methods (Section 5.1), i.e., to mask the mixture magnitude spectrum while using the mixture phase for the reconstruction of time-domain signals. This allows us to compare *direct* separation (using network outputs for the first channel of a microphone array) and separation through beamforming. In Table 5.3, we show results in which the underlying mask predictor was trained either with IBM or IRM targets. Although beamforming (using either mean or product pooling of masks) provides better STOI scores, direct outputs yield considerably better mean SDRi. These results might seem non-intuitive, suggesting that multi-channel processing is not worth exploring. In order to avoid confusion, we stress the dependence of beamforming performance on frame length, as evidenced by Figure 5.5. Following our previous decision, the beamforming in Table 5.3 uses 128 ms long segments[8]. To further emphasize the limitation of direct separation, we show the results of the beamformer and direct separation across various frame lengths in Table 5.4. While the performance of single-channel separation fluctuates, beamforming results steadily improve. Since the analysis is upper-bounded by frames of 192 ms, longer windows would likely reveal a superiority of beamforming, as measured by separation metrics. Moreover, an important aspect is that, in this work, the final goal is to perform speaker verification. Using direct outputs might have a detrimental effect on verification compared to beamforming. This topic will be discussed in Section 6.4.5.

### 5.6.2 Indirect Mask Prediction for Mask-Based Beamforming

An indirect mask prediction demonstrated by Figure 5.4b took inspiration from Beam-TasNet [Ochiai et al., 2020], where a combination of time-domain speaker separation (providing intermediate representations) and beamforming was employed for both separation and ASR. This hybrid solution stemmed from the benefits of beamforming and the rising

[8]For example, we note that the authors set the frame length to 512 ms in [Ochiai et al., 2020] based on their performance analysis.

Table 5.4: Dependence of direct (single-channel) separation and beamforming on frame length. The underlying mask predictor was trained with the IRM targets. The product pooling was used to obtain a global mask for beamforming.

| | **mean SDRi [dB]** | | | | **STOI** | | | |
|---|---|---|---|---|---|---|---|---|
| frame length | 32 ms | 64 ms | 128 ms | 192 ms | 32 ms | 64 ms | 128 ms | 192 ms |
| beamforming | 2.22 | 3.33 | 4.74 | 5.50 | 0.851 | 0.869 | 0.888 | 0.899 |
| direct | 6.10 | 5.49 | 5.80 | 5.46 | 0.874 | 0.869 | 0.872 | 0.868 |

superiority of time-domain speaker separation models established by [Luo and Mesgarani, 2019]. Although indirect mask prediction was discussed in [Ochiai et al., 2020], it was not used due to its inferiority compared to an alternative we will discuss later. However, our original work [Mošner et al., 2022a] revealed the utility of indirect mask prediction. Albeit inspired by a discussion in [Ochiai et al., 2020], our approach to indirect mask prediction is slightly different, and we will present it in this section.

The approach presented here can be thought of as an alternative to (5.10), where time-frequency masks are obtained with the help of time-domain separation models. They predict preliminary representations of speech and noise in the time domain. The signals are then transformed by STFT to produce $\hat{\mathbf{X}}_n$ and $\hat{\mathbf{V}}_n \in \mathbb{C}^{T\times F}$, respectively. Given the estimates of sources, we compute channel-dependent real ratio masks resembling ideal ratio masks (IRM) in (5.2) as

$$\mathbf{M}_n^{(X)} = \left(\frac{|\hat{\mathbf{X}}_n|^2}{|\hat{\mathbf{X}}_n|^2 + |\hat{\mathbf{V}}_n|^2}\right)^{\beta}, \qquad \mathbf{M}_n^{(V)} = \left(\frac{|\hat{\mathbf{V}}_n|^2}{|\hat{\mathbf{X}}_n|^2 + |\hat{\mathbf{V}}_n|^2}\right)^{\beta}, \tag{5.13}$$

where $\beta$ is a scalar set to 0.5 since we empirically found it performing the best. Building upon our previous experiments, we focus on combining the per-channel masks only with the product pooling. Eventually, the rest of the beamformer weights computation is equivalent to that in Section 5.6.1, i.e., SCMs are estimated by (5.10) and the beamformer by (3.63).

### Experiments

#### Direct Outputs of the Networks

In our experiments, we employ two time-domain models, Conv-TasNet and DPTNet, noting that any signal-level separation network can be used. Before diving into beamformer capacity exploration, we will analyze the models themselves. Since they were trained on single-channel data, we can evaluate their separation capability as in the previous section. To this end, we computed mean SDRi and STOI using the first microphone in the array. Table 5.5 (direct method) presents the obtained results. Comparing the metrics with mask-based direct separation (in Table 5.3), we confirm the superiority of time-domain models even on the MultiSV dataset, which has not so far been used in the literature for this purpose. Focusing just on the time-domain models, we observe that DPTNet outperforms Conv-TasNet.

#### Single-Channel vs. Multi-Channel Separation

Next, we focus on the mask-based beamforming utilizing signal-level source predictions. Table 5.5 (mask-based) shows that mean SDRi results are relatively worse than those attained with direct outputs. We also made this observation when analyzing approaches

Table 5.5: Comparison of direct (single-channel) source separation using outputs of the networks and beamforming with the product pooling of per-channel masks. The frame length for beamforming is 128 ms, and the frame shift is 32 ms.

| | **Conv-TasNet** | | | **DPTNet** | | |
|---|---|---|---|---|---|---|
| method | beamforming | | direct | beamforming | | direct |
| SCM estimation | mask-based | mask-less | ✗ | mask-based | mask-less | ✗ |
| mean SDRi [dB] | 6.01 | 6.72 | **9.68** | 6.07 | 6.76 | **10.20** |
| STOI | 0.886 | 0.876 | **0.889** | 0.884 | 0.875 | **0.892** |

with direct mask predictions. Here, the frame length is the same (128 ms); therefore, the outcome is unsurprising. What we consider positive is the fact that the capacity of time-domain models translates to better mask prediction and, hence, the separation ability of the beamformer. This conclusion follows from comparing the mean SDRi of the beamformer outputs in Tables 5.5 and 5.3. STOI, on the other hand, remained comparable. Comparing beamforming results employing Conv-TasNet or DPTNet, we see that they are very close. Therefore, we conclude that the difference in the direct network outputs (distinguishable by the separation metrics) is insufficient to cause a measurable difference in the beamformers' outputs.

### 5.6.3 Mask-Less Beamforming

As indirect mask prediction for mask-based beamforming is an alternative to (5.10) with time-domain separation models used to obtain masks, mask-less beamforming can be viewed as an alternative to (5.12). Here, we aim to employ predicted source signals to estimate SCMs without the intermediate step of computing masks. This approach is similar to that employed in experiments in [Ochiai et al., 2020].

Similarly to the previous section, speech and noise signals predicted by the time-domain models are transformed by STFT to representations $\hat{\mathbf{X}}_n$ and $\hat{\mathbf{V}}_n \in \mathbb{C}^{T\times F}$, respectively. Then, the SCMs are estimated as

$$\mathbf{\Phi}_f^{(xx)} = \frac{1}{T}\sum_{t=1}^{T} \hat{\mathbf{x}}_{t,f}\hat{\mathbf{x}}_{t,f}^{\mathsf{H}}, \qquad \mathbf{\Phi}_f^{(vv)} = \frac{1}{T}\sum_{t=1}^{T} \hat{\mathbf{v}}_{t,f}\hat{\mathbf{v}}_{t,f}^{\mathsf{H}}. \tag{5.14}$$

The difference between this approach and that in [Ochiai et al., 2020] resides in estimating the noise SCM. While we make use of the transformed predicted source $\hat{\mathbf{v}}_{t,f}$, [Ochiai et al., 2020] used $(\mathbf{y}_{t,f} - \hat{\mathbf{x}}_{t,f})$ instead. The latter seems to be more general, as it is more realistic to expect that other sources might be present in a room. On the other hand, it expects that the observed signal is a direct sum of constituent components, which might not hold in reality. Therefore, both methods have their drawbacks. We empirically found that our solution worked better in our particular case and will stick with it. However, it does not mean it is better across all datasets and tasks.

#### Experiments

The mask-less beamformer in our experiments employs the backbone time-domain separation models, which are also used for indirect mask prediction for mask-based beamforming. Therefore, the results are readily comparable. Analyzing the results in Table 5.5, we note

the same behavior as in the previous section. Specifically, despite backbone models providing different mean SDRi and STOI results, the performance is the same after beamforming. Another observation we make is that mask-less beamforming provides better mean SDRi but worse STOI compared to mask-based beamforming with time-domain models.

We should emphasize that these results and results in [Ochiai et al., 2020] were obtained with simulated data. Later, we will revisit all our approaches in the context of pre-processing for multi-channel SV and more realistic (retransmitted) data. At this point, we only note that conclusions about superiority might change when models are confronted with data coming from more difficult acoustic conditions.

## 5.7 Multi-Channel Speech Separation with Cross-Attention and Beamforming

The source separation models presented in Section 5.5.2 and used as a backbone in our previous experiments are single-channel by design. When dealing with beamforming weights prediction, sources need to be predicted at each microphone, which results in a requirement to apply per-channel processing. Hence, channels are fused only later in the pipeline (during beamforming). Per our observation, some sensors in ad-hoc microphone array settings might provide very low SNR outputs. For such microphones, source prediction is erroneous and negatively affects the overall output of the enhancement module (due to having a detrimental impact on the beamforming weights prediction).

Since other channels in the array might be less corrupted, we hypothesize that taking them into consideration already in the neural network-based separation stage might be beneficial. Therefore, as an alternative to per-channel processing, we explore the separation of the reference microphone signal conditioned on all other channels (Figure 5.4b with conditioning). By doing so, we allow the model to recover information that has been corrupted by using signals from other microphones.

In our original work [Mošner et al., 2023], we designed an approach to building multi-channel source separation models by extending current strong single-channel time-domain separators (see Figure 5.6). Our approach allows reusing pre-trained models by inserting a designed lightweight *reference channel attention (RCA) combiner*, the only trained module. It allows for efficient training (especially when the number of microphones is large) as the output error is not back-propagated to the network preceding our module. RCA comprises two blocks: 1) The former allows attending to different parts of other channels w.r.t. the reference one. It was motivated by the potential ability to align information from various channels and, eventually, recover the information occluded in the reference channel from other ones. 2) The latter performs an attention-based combination of channels.

The proposed approach is compatible with various time-domain models with the high-level structure presented in Section 5.5.2 and also shown in Figure 5.3a (such as Conv-TasNet [Luo and Mesgarani, 2019], DPRNN [Luo et al., 2020], DPTNet [Chen et al., 2020], Sepformer [Subakan et al., 2021], and others). Moreover, it allows us to conveniently estimate beamforming weights and perform a combination of channels through spatial filtering. In our experiments, we will emphasize the synergy of multi-channel signal processing (beamforming) and network-based separation (especially when using more microphones).

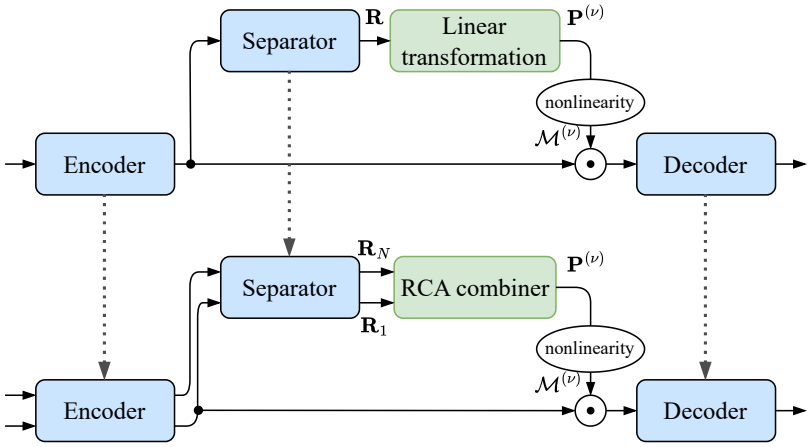



Figure 5.6: Extension of single-channel source separation models to multi-channel ones by replacing the linear transformation with the reference channel attention (RCA) combiner. Dotted arrows represent a transfer of pre-trained weights. Linear transformation and the RCA combiner are specific to the source $\nu \in \{X, V\}$. We show only one of them for clarity.

### 5.7.1 Reference Channel Attention Combiner

When describing Reference Channel Attention (RCA), we will rely on the structure of time-domain models described in Section 5.5.2. We recall that within the structure of time-domain single-channel separation models [Luo and Mesgarani, 2019, Luo et al., 2020, Subakan et al., 2021], the *separator base* block (shown in Figure 5.3a) produces the representation $\mathbf{R} \in \mathbb{R}^{\mathfrak{T} \times \mathfrak{D}}$, where $\mathfrak{T}$ represents total number of frames (provided by the encoder) in the separation stage and $\mathfrak{D}$ is the dimensionality of the network feature space. Importantly, $\mathbf{R}$ is linearly projected to per-source features:

$$\mathbf{P}^{(X)} = \mathbf{R}\mathbf{W}^{(X)}, \quad \mathbf{P}^{(V)} = \mathbf{R}\mathbf{W}^{(V)}. \tag{5.15}$$

The projection matrices $\mathbf{W}^{(\cdot)} \in \mathbb{R}^{\mathfrak{D} \times D}$ are unique for the sources $X$ (speech) and $V$ (noise). The dimensionality of the output features is $D$. Masks for individual sources are then obtained as[9]

$$\mathcal{M}^{(X)} = \varsigma(\mathbf{P}^{(X)}; \theta), \quad \mathcal{M}^{(V)} = \varsigma(\mathbf{P}^{(V)}; \theta), \tag{5.16}$$

with $\varsigma$ representing non-linearity (e.g., sigmoid, ReLU) or more complex non-linear function parametrized by $\theta$ [Chen et al., 2020].

#### Extending Single-Channel Models

This section describes a straightforward extension of existing pre-trained source separation networks to multi-channel ones that provide outputs aligned with the reference microphone as illustrated in Figure 5.6. The alignment is important 1) to be able to compute time-domain loss correctly, 2) since it allows obtaining predictions for all channels by changing the reference one (it will be used in the beamforming weights estimation).

[9]Note that we use a different symbol for masks within separating networks compared to those used in the mask-based beamforming to make a clear distinction between the two.

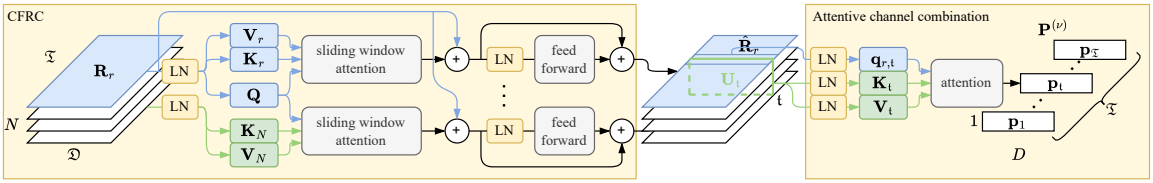



Figure 5.7: Reference channel attention (RCA) module fuses input multi-channel representations $\mathbf{R}_n$ w.r.t. the reference channel $r$ (in blue) to yield the output representation $\mathbf{P}^{(\nu)}$. LN stands for layer normalization.

Rows of $\mathbf{R}$ (i.e., vectors $\mathbf{r}_t$) live in such a space that a linear transformation projects them to source-specific representations. Therefore, we hypothesize that $\mathbf{R}$ represents a suitable intermediate internal output where the information from microphones can be fused.

A simple approach would be to average channel-specific $\mathbf{R}_n$. However, due to sound propagation delay, misalignment could occur. It would result in unwanted smoothing. Instead, we propose the RCA module that allows for attending to different parts of different channels to improve the estimation of source signals at the reference microphone. Our framework reuses the pre-trained model to predict $\mathbf{R}_n$ for all channels $n = 1, \ldots, N$. Subsequently, the RCA module substitutes linear transformation in (5.15) as follows:

$$\mathbf{P}^{(\nu)} = \mathcal{F}_{\text{RCA}}\left(\{\mathbf{R}_n\}_{n=1}^{N}; \phi^{(\nu)}\right), \tag{5.17}$$

with $\mathcal{F}_{\text{RCA}}$ representing the RCA combiner parametrized by source-specific weights $\phi^{(\nu)}$. Subsequent computation steps follow the original architecture into which the module is inserted.

### Description of RCA

A diagram of the RCA combiner corresponding to one source is depicted in Figure 5.7. It comprises two modules: *cross-frame reference channel attention* (CFRC, partially resembling [Chang et al., 2021], but modified to fit the purpose of extracting sources at the reference channel) and *attentive channel combination* (ACC, similar to [Wang et al., 2020]). To simplify the description, we will omit normalization. We emphasize that the description in the following text is for one source $\nu$, noting that an analogy holds for the second source. We will stress it by explicitly using the symbol $\nu$ at the very end of the description. Otherwise, we refrain from using it to simplify notation.

**Cross-frame reference channel attention:** The CFRC enables the model to attend to different parts of different channels, where the goal is to improve the estimation of source signals at the reference microphone (marked in blue). It is achieved by the attention design that uses common *queries* $\mathbf{Q} = \mathbf{R}_r\mathbf{W}^{(Q)} \in \mathbb{R}^{\mathfrak{T}\times D}$ extracted from the reference microphone denoted by subscript $r$. Channel-specific *keys* and *values* are obtained by projecting $\mathbf{R}_n$ with weight matrices that are shared across channels:

$$\mathbf{K}_n = \mathbf{R}_n\mathbf{W}^{(K)} \in \mathbb{R}^{\mathfrak{T}\times D}, \tag{5.18}$$

$$\mathbf{V}_n = \mathbf{R}_n\mathbf{W}^{(V)} \in \mathbb{R}^{\mathfrak{T}\times D}. \tag{5.19}$$

We note that the weight-sharing approach makes the module agnostic to the number of sensors.

In the following step, depending on the channel, self- or cross-attention is performed. The initial implementation in [Mošner et al., 2023] used relative positional encoding [Shaw et al., 2018]. In our experiments, we will show its utility. We also omit it from formulas for brevity. By utilizing a sliding window version of the attention mechanism [Luong et al., 2015, Beltagy et al., 2020], we allow the query frame to attend to keys from a limited time context. The rationale behind this choice is twofold: First, the CFRC attention module should be able to align the information from various microphones. Therefore, a limited context is sufficient, as the signal shift is limited (by the speech of sound and array size). Second, it is a memory-saving solution — the resulting system can process input sequences of various lengths without the problem of quadratic memory growth with the length of the sequence [Vaswani et al., 2017].

In order to formalize it (without layer normalization and positional encoding), let $\mathbf{q}_\mathfrak{t} \in \mathbb{R}^D$ be a query vector corresponding to the frame $\mathfrak{t}$[10] (that is, the $\mathfrak{t}$th row of $\mathbf{Q}$). Analogically, let vectors $\mathbf{k}_{n,\mathfrak{t}} \in \mathbb{R}^D$ and $\mathbf{v}_{n,\mathfrak{t}} \in \mathbb{R}^D$ be $\mathfrak{t}$th rows of $\mathbf{K}_n$ and $\mathbf{V}_n$, respectively. Then the output $\mathbf{a}_{n,\mathfrak{t}}$ of the sliding window attention for frame index $\mathfrak{t}$ and channel $n$ is

$$\begin{aligned} \mathbf{a}_{n,\mathfrak{t}}^\intercal &= \mathrm{softmax}\left(\frac{\mathbf{q}_\mathfrak{t}^\intercal \mathbf{K}_{n,\mathcal{S}_\mathfrak{t}}}{\sqrt{D}}\right)\mathbf{V}_{n,\mathcal{S}_\mathfrak{t}}^\intercal, \\ \mathbf{K}_{n,\mathcal{S}_\mathfrak{t}} &= (\mathbf{k}_{n,\tau})_{\tau\in\mathcal{S}_\mathfrak{t}} \in \mathbb{R}^{D\times(2\delta+1)}, \\ \mathbf{V}_{n,\mathcal{S}_\mathfrak{t}} &= (\mathbf{v}_{n,\tau})_{\tau\in\mathcal{S}_\mathfrak{t}} \in \mathbb{R}^{D\times(2\delta+1)}. \end{aligned} \tag{5.20}$$

$\mathcal{S}_\mathfrak{t} = \{\mathfrak{t}-\delta,\dots,\mathfrak{t}+\delta\}$ is a set of frame indices around $\mathfrak{t}$ within a context of $\pm\delta$.

Vectors $\mathbf{a}_{n,\mathfrak{t}}$ are eventually concatenated over time and subsequently summed with $\mathbf{R}_r$ through a residual connection. The summed representations are passed to a block with layer normalization, a feed-forward layer, and a residual connection to produce a multi-channel output of the CFRC $\{\hat{\mathbf{R}}_n\}_{n=1}^N$.

**Attentive channel combination:** The last stage of the RCA module, ACC, is supposed to combine the aligned channels in a per-frame manner (being invariant to the number of channels). To this end, we need to consider concatenated outputs of the CFRC module: $[\hat{\mathbf{R}}_1, \hat{\mathbf{R}}_2, \dots, \hat{\mathbf{R}}_N] \in \mathbb{R}^{N\times\mathfrak{T}\times D}$. Then, the same processing steps can be repeated for every frame index (i.e., slice $\mathbf{U}_\mathfrak{t} \in \mathbb{R}^{N\times D}$ of the aforementioned tensor). We note that in practice, computation is performed in parallel thanks to independence

As detailed in Figure 5.7, the combination is performed w.r.t. the reference channel (marked in blue). A query vector is extracted only from the reference channel $r$: $\mathbf{q}_{r,\mathfrak{t}} = \mathbf{W}^{(Q,\mathrm{ACC})}\mathbf{u}_{r,\mathfrak{t}} \in \mathbb{R}^D$, where $\mathbf{u}_{r,\mathfrak{t}} \in \mathbb{R}^D$ is the row of $\mathbf{U}_\mathfrak{t}$ corresponding to the reference microphone $r$. The output of the block is

$$\mathbf{p}_\mathfrak{t} = \mathrm{softmax}\left(\frac{\mathbf{q}_{r,\mathfrak{t}}^\intercal\left(\mathbf{U}_\mathfrak{t}\mathbf{W}^{(K,\mathrm{ACC})}\right)^\intercal}{\sqrt{D}}\right)\left(\mathbf{U}_\mathfrak{t}\mathbf{W}^{(V,\mathrm{ACC})}\right), \tag{5.21}$$

where $\mathbf{W}^{(Q,\mathrm{ACC})}$, $\mathbf{W}^{(K,\mathrm{ACC})}$, and $\mathbf{W}^{(V,\mathrm{ACC})}$are projection matrices of shape $D\times D$. We note that $\mathbf{p}_\mathfrak{t}$ can be viewed as a weighted average (linear combination) of channel-specific values. The vectors $\mathbf{p}_\mathfrak{t}$ are finally concatenated, yielding a source-specific matrix $\mathbf{P}^{(\nu)} \in \mathbb{R}^{\mathfrak{T}\times D}$ which is an alternative to that in (5.15).

We note that this module is a version of *multi-head attention across channels* [Wang et al., 2020] (also used in [Horiguchi et al., 2022b]). In contrast to [Wang et al., 2020], a query is extracted only from the reference channel (not from all).

[10] Note that $\mathfrak{t}$ and $t$ denote network and the T-F domain frames, respectively.

### 5.7.2 Extension of Well-Known Single-Channel Separation Models

Although the RCA combiner is compatible with all models following the structure depicted in Figure 5.3a, specific implementation details of individual models may result in adjustments. Since the base models of our focus are Conv-TasNet and DPTNet, we will detail how to integrate RCA into them.

#### Conv-TasNet

Insertion of the RCA combiner into Conv-TasNet follows Figure 5.6 exactly. The linear transformation in (5.15) is commonly implemented by 1×1 convolution[11]. Hence, our module seamlessly replaces the convolutional layer in the original model.

#### DPTNet

As mentioned before, in the original DPTNet [Chen et al., 2020], linear transformation in the separator block (Figure 5.3a) is applied to all features vectors of all segments that were introduced by chunking required by dual-path processing[12]. Then, the segments are combined in an overlap-add fashion. To reduce the computational and memory burden associated with a simple replacement of the linear transformations (and also to conform to a description of the general structure provided in Section 5.5.2), we first merge segments via overlap-add (see Figure 5.3c). This yields $\mathbf{R}_n$ — suitable input to the RCA combiner. We argue that this is a minimal change, having the capacity to provide the same outputs as the original design.

### 5.7.3 Experiments With Reference Channel Attention

#### Ablation of the Reference Channel Attention Combiner

In this section, we focus on the RCA combiner design choices and provide an ablation study of the module elements. We chose one specific architecture of the separation network that we augmented with the RCA and evaluated it on the MultiSV *eval simu* set. As we are not interested in the absolute performance but rather in relative differences brought by architectural changes, we opted for the DPTNet low (see Table 5.1) separation model (which also saves training time).

Arguably, the strength of the RCA lies in the CFRC since it is the only module capable of dealing with time misalignment between channels. In order to confirm this hypothesis, we tested the separation performance with the CFRC removed from the RCA. Then, the channel combination module comprises only the ACC (see Figure 5.7). It is worth noting that the resulting network degrades to a single-channel one under two assumptions: 1) when the weight matrix $\mathbf{W}^{(V,\mathrm{ACC})}$ transforming the internal representations to values equals the projection matrix $\mathbf{W}^{(\nu)}$ in (5.15), 2) the attention mechanism always selects the reference channel. Since the latter is not viable due to the correlation of channel signals, the model with the RCA comprising only ACC will likely mix misaligned representations of the reference and other channels (compared to a single-channel model). As shown in

[11] For example, in the official implementation (https://github.com/naplab/Conv-TasNet) and Asteroid, a toolkit for source separation research: (https://github.com/asteroid-team/asteroid).

[12] This scheme is followed in the official implementation (https://github.com/ujscjj/DPTNet) and also in Asteroid.

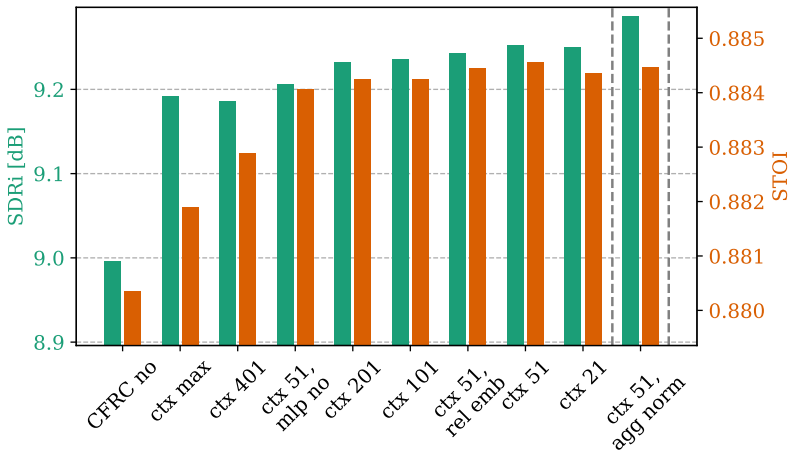



Figure 5.8: Impact of the RCA hyperparameters on the separation performance.

Figure 5.8, the corresponding model, termed *CFRC no*, is indeed the worst out of all the variants.

Having verified the utility of the CFRC, we kept it in place and explored the effect of its sliding window attention context. We started with the standard attention (*ctx max*), having no constraints on the context and attending to the whole sequence. Then, we gradually decreased the context to 401, 201, 101, 51, and 21 ms. The corresponding models are referred to as *ctx {401, 201, 101, 51, 21}* in Figure 5.8, respectively. We note that the context corresponds to the entire set $\mathcal{S}_t$ in (5.20). Although differences are limited, an improving trend of separation metrics is observable. Allowing the CFRC to attend to the full history and future tends to be confusing. Better performance with limited context suggests that the cues useful for separation do not spread far (in terms of frames). It corresponds to the intuition that time locality correlates with space locality. We note that the optimal context depends on a spatial configuration of the actual operational environment. Based on the experiments, we opt for 51 ms (*ctx 51*) out of the limited contexts and keep it fixed in our further experiments.

In the next phase, we explored other architectural changes. Following [Mošner et al., 2023], we incorporated relative positional encoding into CFRC (*ctx 51, rel emb*). Since we observed no benefit, we did not further consider it. In [Liang et al., 2022], the authors showed improvements stemming from removing a position-wise feed-forward network from the Transformer encoder block. Along the same lines, we removed it from the CFRC (*ctx 51, mlp no*). However, we did not make the same observation; therefore, we kept the feed-forward network in place. Finally, following the common practice of applying layer normalization in a pre-norm scheme, we introduced layer normalization to the ACC module (in contrast to [Mošner et al., 2023]). The corresponding model is termed (*ctx 51, agg norm*). This modification yielded the best version and is used in our further experimentation.

Overall, the architecture that turned out to be the most performant differs from its original version we introduced in [Mošner et al., 2023] as follows. It comprises layer normalization in the ACC, and the sliding window attention in the CFRC allows attending to a limited context whose length was tuned. On the other hand, it lacks positional encoding in the CFRC.

Table 5.6: Comparison of the RCA training alternatives and mask-based MVDR beamforming using a frame size of 128 ms. The separation metrics were computed on MultiSV *eval simu*. The *RCA* column indicates the utilization of the reference channel attention module. When the *RCA adapt.* is on, the whole multi-channel separation model, including the RCA, was adapted on multi-channel data.

| Base model | | RCA | RCA adapt. | direct output | | beamforming | | fusion | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | mean SDRi | STOI | mean SDRi | STOI | mean SDRi | STOI |
| FaSNet TAC | A | ✗ | ✗ | 9.73 | 0.889 | — | — | — | — |
| Conv-TasNet | B | ✗ | ✗ | 9.68 | 0.889 | 6.01 | 0.886 | 9.61 | 0.912 |
| | C | ✓ | ✗ | 9.70 | 0.892 | 6.01 | 0.886 | 9.62 | 0.914 |
| | D | ✓ | ✓ | 9.85 | 0.894 | 5.96 | 0.885 | 9.67 | 0.915 |
| DPTNet | E | ✗ | ✗ | 10.20 | 0.892 | 6.07 | 0.884 | 9.89 | 0.914 |
| | F | ✓ | ✗ | 10.24 | 0.894 | 6.05 | 0.884 | 9.89 | 0.916 |
| | G | ✓ | ✓ | 10.24 | 0.895 | 6.04 | 0.885 | 9.89 | 0.917 |

### RCA Combiner Training Strategies and Mask-based Beamforming

In the previous experiment with RCA and the initial ones in this section, we followed our initial training strategy presented in [Mošner et al., 2023]. In the first stage, a time-domain separation network was trained on single-microphone data, taking a random channel from the MultiSV training examples in our case. Models B and E in Table 5.6 represent two such instances with different backbone architectures. Then, we incorporated a randomly initialized RCA block into the architecture, making the model capable of processing multi-channel data. We kept all the parameters fixed except for those of RCA (as illustrated in Figure 5.6). The trainable weights were updated using multi-channel MultiSV training data, keeping the same loss function defined in (5.11). This step was adopted by models C and F, employing B and E as backbones, respectively. In this section, we will extend the outlined approach. Given that models C and F are multi-channel, we can further fine-tune them by unfreezing all the parameters and training on the MultiSV training data. This strategy provided models D and G.

First, we will concentrate on the *direct output* of the abovementioned networks. Focusing on models B and E, we observe an unsurprising increase in the separation metrics by employing a more advanced model. The trend in results reflects progress in the source separation field. Departing from single-channel models (B, E), we observe a slight improvement from employing more channels (models C, D, and F, G) on simulated data. Nevertheless, our multi-channel versions of models can outperform FaSNet TAC [Luo et al., 2019] trained on the same data.

Next, we will consider using the separated time-domain outputs as intermediate representations that will be used to estimate beamforming weights. We will follow the MVDR mask-based beamforming with indirectly predicted masks as presented in Section 5.6.2. The indirect prediction using models B and E follows Figure 5.4b without conditioning. On the other hand, models employing RCA correspond to Figure 5.4b with conditioning (where channels other than the reference one are considered as conditioning). In Section 5.6.2, we observed that the performance of mask-based beamforming employing frames of 128 ms is considerably worse compared to direct network outputs. The results in the *beamforming* section of Table 5.6 confirm these trends. Moreover, a limited difference in the direct output results has a minor impact on the difference in the beamformer results. While the utility of beamforming might seem questionable, we recall the dependence on the STFT

Table 5.7: The results in this table correspond to those in Table 5.6. The difference is that a frame size of 256 ms was used instead of 128 ms.

| Base model | | RCA | RCA adapt. | beamforming | | fusion | |
|---|---|---|---|---|---|---|---|
| | | | | mean SDRi | STOI | mean SDRi | STOI |
| Conv-TasNet | B | ✗ | ✗ | 8.09 | 0.912 | 10.52 | 0.922 |
| | C | ✓ | ✗ | 8.04 | 0.911 | 10.50 | 0.923 |
| | D | ✓ | ✓ | 8.01 | 0.911 | 10.54 | 0.923 |
| DPTNet | E | ✗ | ✗ | 8.21 | 0.912 | 10.87 | 0.922 |
| | F | ✓ | ✗ | 8.16 | 0.911 | 10.84 | 0.924 |
| | G | ✓ | ✓ | 8.12 | 0.912 | 10.80 | 0.925 |

window size. In order to demonstrate the effect of frame size in the context of time-domain separators, we show beamforming results following the same setup as before while increasing the frame size from 128 ms to 256 ms in Table 5.7. We observe a clear improvement. Comparing the beamforming result with the direct outputs, we conclude that although the mean SDRi of the beamformed signals is still worse, the STOI results show superiority of the beamforming. At this point, we also note that another important factor affecting the performance of beamformers is the number of microphones (which was already discussed in Section 3.1.2). Although we do not explore it here, we will tackle the issue in the context of both separation and speaker verification in Section 6.4.5.

Drawing inspiration from the benefits brought by the fusion of the beamformer and network outputs in source separation that we presented in [Mošner et al., 2023], we conducted analogous output combination experiments in our scenario. Given speech prediction $\hat{\mathbf{X}}^{(BF)}$ provided by the beamformer according to (3.62), the time-domain representation $\hat{\mathbf{x}}^{(BF)}$ can be obtained by inverse STFT. By exchanging speech and noise SCMs in the beamformer, we can obtain the prediction of noise $\hat{\mathbf{V}}^{(BF)}$ and hence $\hat{\mathbf{v}}^{(BF)}$[13]. Recalling that time-domain separators provide estimates of speech $\hat{\mathbf{x}}$ and noise $\hat{\mathbf{v}}$, a simple fusion approach allowed by not enforcing scale invariance in training, i.e., by using (5.11), is given by

$$\hat{\mathbf{x}}^{(\text{fus})} = \frac{\hat{\mathbf{x}}^{(BF)} + \hat{\mathbf{x}}}{2}, \quad \hat{\mathbf{v}}^{(\text{fus})} = \frac{\hat{\mathbf{v}}^{(BF)} + \hat{\mathbf{v}}}{2}. \tag{5.22}$$

We explicitly express $\hat{\mathbf{v}}^{(\text{fus})}$, noting that it participates only in the computation of the mean SDRi metric. When evaluating STOI, only $\hat{\mathbf{x}}^{(\text{fus})}$ is used. Table 5.6 shows that the *fusion* improved results compared to mere beamforming. Focusing on the direct outputs, we can conclude that although the mean SDRi of the fused outputs is worse, the fusion brought improvements in speech intelligibility as measured by STOI. Since we previously obtained beamformed outputs using frames of 256 ms, it is straightforward to perform fusion of them and the direct outputs as well. Table 5.7 demonstrates that an improved beamforming performance translates to better fusion results. The fusion eventually yields better results compared to even direct outputs in all measure metrics. We can therefore summarize that if the beamformer mean SDRi is not significantly worse than that of direct outputs, an expectation of improvement is justified.

Although this section presented different ways of training RCA, the direct output results might seem unconvincing regarding the RCA contribution. However, we note in advance

[13] We note that sources of noise are point sources in our data.

that the RCA becomes interestingly useful when using it in the pre-processing for speaker embedding extraction for SV. In that case, it processes more realistic (i.e., retransmitted) data compared to simulated ones. The analysis will be provided in Section 6.4.5.

## 5.8 Summary

This chapter laid the basis for front-ends used later in our cascaded multi-channel speaker embedding extractor versions. We initially reviewed approaches to single- and multi-channel separation/enhancement that influenced our pre-processing models. Next, we introduced standard metrics to evaluate separation performance and detailed those we employed in our experiments.

We devoted the rest of the chapter to analyses and experiments. Prior to delving into approaches employing neural networks, we focused on the impact of the type of ideal masks and pooling, yielding one global mask per source. These are used when estimating beamforming weights. We evaluated the impact of the mask and pooling types for various frame sizes used when transforming signals with STFT. We observed an improving trend in separation metrics with increased frame size. We also concluded the superiority of ideal ratio masks when aggregated with the product pooling.

Then, we proceeded with approaches that build on separation neural networks. We presented three methods for multi-channel pre-processing using separators and beamforming: direct and indirect mask prediction for mask-based beamforming and mask-less beamforming. In the context of direct mask prediction, we concluded that the IRM training targets lead to better-performing models (compared to the IBM targets), confirming the utility of ratio masks. In accordance with the oracle experiments, product pooling turned out to be the best aggregation method. Making use of time-domain separation models, we were concerned with indirect mask prediction for mask-based beamforming. Having gained insights about the effects of masks, we employed ratio masking given the STFT images of predicted signals of sources that we eventually aggregated by product pooling. Our experiments revealed that indirect mask prediction using strong time-domain separators helps beamforming attain better separation performance (compared to direct mask estimation). Next, we employed the time-domain models to estimate SCMs (required by beamformers) without the need for masks (i.e., mask-less beamforming). While the technique led to better SDRi compared to mask-based beamforming, the intelligibility of the outputs did not match that achieved with mask-based beamforming.

We dedicated the last part of the chapter to our technique, the reference channel attention (RCA) combiner, which extends various single-channel time-domain separation networks to multi-channel ones. We performed ablation experiments that shed light on the effect of module components. Then, we discussed strategies for the RCA training. Time-domain separators incorporating the RCA predict single-channel signals of sources given a multi-channel mixture input. Therefore, they can be used directly or to support beamforming following the indirect-mask prediction scheme. Eventually, the direct outputs and the outputs of the beamformer can be fused. We were interested in comparing such approaches to the prediction of sources. The outcomes of this chapter directly impact front-ends employed in Chapter 6, where we will revisit them in the context of the multi-channel SV task.

# Chapter 6

# Multi-channel Speaker Verification

Multi-channel speaker verification is the core task of our work. This chapter will concentrate on the cascaded approach integrating multi-channel pre-processing and a single-channel verification module. When developing the pre-processing module, we will employ key findings from the previous chapter. We will aim to converge to fully differentiable model instances that can be fine-tuned on the multi-channel data with the speaker-discriminative loss.

We will start by defining the task of multi-channel speaker verification and reviewing approaches from the literature. Then, we will outline our work. First, we will focus on reverberant data and dealing with them in the context of the system employing i-vector representations and a PLDA backend. We presented this work in [Mošner et al., 2018a, Mošner et al., 2018b]. Then, we will establish a general framework of beamforming-based pre-processing and embedding extraction by neural networks, which we developed in a series of papers [Mošner et al., 2019, Mošner et al., 2022a, Mošner et al., 2025]. Here, we will explore proof-of-concept investigations followed by a thorough analysis in light of insights gained in previous separation experiments. Finally, we will present a strategy for fine-tuning all parameters with a single objective, leading to performance gains.

## 6.1 Task Definition

Multi-channel speaker verification is a special case of general speaker verification. Therefore, it follows the definition of verification as provided in Section 2.1. Likewise, multi-channel SV systems are evaluated on trial lists comprising target and non-target (impostor) examples, allowing for the computation of metrics defined in Section 2.2. The main specificity relates to the number of channels (signals captured by microphone arrays) in enrollment and test. We emphasize that channels are specifically constrained: they contain different versions of the same audio content captured by different microphones in different spatial locations, making them statistically dependent. Therefore, this scenario should not be confused with the multi-session enrollment scenario common in general SV, where channel independence is expected, for example, in by-the-book PLDA scoring [Lee et al., 2013].

Multi-channel SV approaches can be categorized by the number of channels in the enrollment and test parts of verification trials:

**Single-channel enrollment, multi-channel test** [Cai and Li, 2021, Qin et al., 2023, Mošner et al., 2022a] A practically interesting scenario assuming that the user is enrolled with a single recording, typically recorded by a close-talking device. At the evaluation time, recordings from microphone arrays are used to extract one embedding

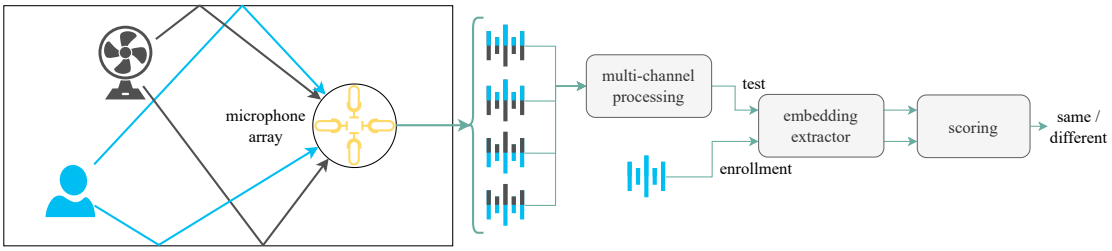



Figure 6.1: A simplified scheme of a version of multi-channel speaker verification systems operating in room conditions. The diagram presents a scenario with single-channel enrollment and multi-channel test.

to be compared with the enrollment embedding. Figure 6.1 shows an illustration of a potential system dealing with the difference of channels in enrollment and test. This scenario is supported by our MultiSV CE and SRE protocols.

**Multi-channel test and enrollment** [Liang et al., 2021,Liang et al., 2022,Mošner et al., 2024] Such a scenario expects that microphone arrays provide both enrollment and test sets of recordings. MultiSV supports it through the MRE and MRE_hard protocols.

One can also think of multi-channel enrollment and a single-channel test case. However, it is not appealing in practice in the context of microphone arrays. Therefore, studies on multi-channel SV usually do not consider it. In our work, we will be concerned with the two main scenarios, and most of the systems we developed are usable in both of them. The exceptions are models fine-tuned as a whole on multi-channel data (Section 6.4.5 and Chapter 7).

## 6.2 Approaches to Multi-Channel Speaker Verification

Despite automatic speech recognition (ASR) being a dominant field since speech transcription is the primary goal in many use cases, the application of microphone arrays in speaker recognition has been a task of interest for years, as supported by published studies. A common feature of early works is that they employ multi-channel signal processing, namely, beamforming. The pioneering speaker recognition work utilizing beamforming dates back to 1994 [Lin et al., 1994]. The authors explored multi-channel signal processing applied to tens of sensor signals. Specifically, the matched-filter microphone array provided enhanced output [Jan et al., 1995]. We can interpret it as the filter-and-sum beamformer, where the matched-filter is dedicated to each microphone. The final task was not SV but speaker identification using the vector quantization (VQ) based speaker recognition system [Soong et al., 1985]. Along the same lines, [Gonzalez-Rodriguez et al., 1996] proposed a multi-channel speaker identification system where low computational complexity was of interest. It extended the previous work by employing the recognition system based on Gaussian mixture model representations of speakers [Reynolds and Rose, 1995] and different beamforming. The authors used a four-microphone switched array and Griffiths-Jim beamformer [Griffiths and Jim, 1982] where channel delays are estimated when speech is present, and adaptation coefficients are updated in pauses [Compernolle et al., 1989]. The first multi-channel text-independent speaker verification system was presented in [McCowan et al., 2001], representing an important milestone. It focused on the following aspects: more

sophisticated beamforming, input data generation, and the effect of multi-channel processing on the SV performance. The employed spatial filtering was near-field adaptive filtering (NFAB) [McCowan et al., 2000].

Despite [McCowan et al., 2001] focusing on more realistic data, all early works generated test data by convolving room impulse responses (RIRs) with clean data. In [McCowan et al., 2001], RIRs were measured in a genuine room. In other studies, they were simulated using the image source method [Allen and Berkley, 1979] built upon the ray acoustics basis. As such, it makes assumptions that are not met in reality. Moreover, room, speech, and noise source locations were unchanged in all the studies, with the distraction signal being white noise, while recent multi-channel SV works stress the test data variability and the need for generalizability of models. Other commonalities are using neural networks in pre-processing (already suggested in [Lin et al., 1994]) and speaker embedding extractors. Recent approaches to multi-channel SV can be categorized as follows: 1) models with pre-processing [Mošner et al., 2018a, Taherian et al., 2019, Yang and Chang, 2019, Taherian et al., 2020, Mošner et al., 2022a, Dowerah et al., 2023b, Dowerah et al., 2023a], 2) models with late fusion [Cai and Li, 2021, Liang et al., 2021, Liang et al., 2022], 3) model-level approaches [Cai et al., 2019, Qin et al., 2023].

**Models with pre-processing:** Following the early success of [McCowan et al., 2001], numerous approaches have built upon signal-level pre-processing. Most studies adopt beamforming to provide a single-channel enhanced signal, further fed to a neural network-based speaker embedding extractor. A usual beamforming technique of choice is either a generalized eigenvalue beamformer (GEV) or MVDR (discussed in Section 3.2). Despite adhering to different criteria, the optimization leads to closed-form solutions with spatial covariance matrices (SCMs) of speech and noise in both cases. The performance of the beamforming depends on the quality of the SCM estimates. Therefore, most studies leverage the modeling power of neural networks for the SCM prediction. Following [Heymann et al., 2016a], which focuses on ASR, numerous works [Mošner et al., 2018a, Yang and Chang, 2019, Taherian et al., 2019] adopt the direct speech and noise mask estimation (following approaches similar to that in Figure 5.4a). Notably, [Taherian et al., 2019] showed that given the same mask prediction, the SV performance can be improved by a rank-1 approximation of a speech SCM [Serizel et al., 2014]. Except for beamforming, we showed in our study [Mošner et al., 2018a] that a preceding dereverberation can help to boost performance. A limitation of approaches with pre-processing (as documented by [Shon et al., 2019] in the single-channel scenario) is that the enhancement objective is not necessarily well-correlated with the final objective. Therefore, joint optimization with the ultimate objective turns out to be promising. Despite the theoretical differentiability of backend and beamforming, gradient computation within a connected model has its pitfalls due to complex-valued operations such as eigen-decomposition [Boeddeker et al., 2017, Mošner et al., 2019]. Training also might get unstable owing to matrix inversion. However, [Yang and Chang, 2019] showed that a successful joint fine-tuning brings noticeable speaker verification performance gains. Apart from beamforming, a multi-channel enhancement approach based on diffusion probabilistic models was designed for SV [Dowerah et al., 2023b]. It was further developed in [Dowerah et al., 2023a], where the Conv-TasNet architecture [Luo and Mesgarani, 2019] was incorporated to perform the diffusion process. The resulting model, Diff-Filter, was trained to provide outputs that resemble outputs of a rank-1 multi-channel Wiener filter (MWF) [Serizel et al., 2014]. The study also emphasized benefits brought by joint optimization with the ECAPA-TDNN embedding extractor [Desplanques et al., 2020].

**Models with late fusion:** Another line of research leverages neural speaker embedding extractors proven strong in single-channel experiments and challenges [Huh et al., 2023]. A usual model of choice is a variant of ResNet [He et al., 2016a]. Similarly to other state-of-the-art embedding extractors, it has a three-stage high-level structure described in Section 2.4.1: frame-level processing, pooling, and utterance-level processing. In multi-channel models, signals from different microphones are processed by replicas of the original single-channel model up to a particular layer. Subsequently, information is exchanged between channels, and what follows is a single-channel processing. Both [Cai and Li, 2021] and [Liang et al., 2021] applied frame-level processing and pooling for each channel and performed information exchange and aggregation at the utterance level. Aggregation of per-channel embeddings through average or attention-based weighted-average was explored in [Cai and Li, 2021]. The study in [Liang et al., 2021] presented an approach where inter-channel processing layers and global fusion layers were incorporated into the embedding extractor after the pooling operation. Both types of layers made use of a cross-channel multi-headed residual attention. On the other hand, [Liang et al., 2022] proposed a model where the multi-channel aggregation module was introduced at the end of frame-level processing (right before the pooling layer). The aggregation utilized either a succession of cross-frame and cross-channel attention or graph attention. It tended to outperform the two mentioned utterance-level approaches at the expense of higher computational complexity.

**Model-level approaches:** While still sharing the same 3-stage high-level structure, model-level approaches aim to accommodate multi-channel data by design. In [Cai et al., 2019], microphone channels were treated as input channels to a convolutional layer. Notably, the authors also proposed an architecture based on 3D (or both 3D and 2D) convolutional layers. The extension with the Squeeze-and-Excitation block [Hu et al., 2020a] designed explicitly for 3D convolutional layers was presented in [Qin et al., 2023].

## 6.3 Multi-Channel Dereverberation and I-vectors

When dealing with far-field data captured in enclosures (such as rooms), reverberation is the fundamental source of problems for speech processing models. Therefore, our initial work in multi-channel SV aimed to quantify the impact of reverberation on speaker verification [Mošner et al., 2018a, Mošner et al., 2018b]. Observing severe degradation, we focused on designing and analysing approaches to combat reverberation, which we will describe in this section. When we developed our systems, the i-vector representation of recordings combined with the PLDA backend was still the most common and performant approach to SV. Therefore, it also underlies our designs.

### 6.3.1 Pre-Processing and Adaptation

In this part, we will focus on two dereverberation techniques differing in their nature. The first is weighted prediction error (WPE) based on delayed linear prediction [Nakatani et al., 2010, Yoshioka and Nakatani, 2012]. It is capable of processing both single- and multi-channel data. The dereverberation/denoising autoencoder [Plchot et al., 2016] will be the second method of our interest.

Apart from dereverberation, we will also aim to perform beamforming. The inherent property of ad-hoc microphone arrays we work with is that no prior information on microphone positions is available. The algorithms processing their outputs based on the direction of arrival need to locate the direction of interest blindly. Various approaches were

developed to tackle this problem [Himawan et al., 2011]. Here, we will use time difference of arrival (TDOA) estimation based on generalized cross-correlation with phase transform (GCC-PHAT) [Knapp and Carter, 1976]. Let us consider our signal model in Section 3.3.1, then the GCC-PHAT between the observed frequency-domain signals of microphones $n_1$ and $n_2$ is given as

$$\hat{R}_{\text{PHAT}}(t, f, n_1, n_2) = \frac{Y_{t,f,n_1} Y^*_{t,f,n_2}}{|Y_{t,f,n_1} Y^*_{t,f,n_2}|} \tag{6.1}$$

A time delay between the two microphones is then estimated as

$$\hat{d}_{\text{PHAT}}(t, n_1, n_2) = \arg\max_{\mathtt{t}} \left\{ \mathcal{F}^{-1}_{\text{DFT}} \left[ \hat{R}_{\text{PHAT}}(t, :, n_1, n_2) \right]_{\mathtt{t}} \right\}, \tag{6.2}$$

where the symbol ":" signifies taking all frequency bins, and discrete time $\mathtt{t}$ indexes the output of the inverse discrete time Fourier transform $\mathcal{F}^{-1}_{\text{DFT}}$. Given the estimated delays, a basic delay-and-sum (DS) beamformer, defined in (3.24), will serve as our spatial filter. As discussed in Section 5.2, using neural networks in acoustic beamforming became widely embraced [Xiao et al., 2016, Heymann et al., 2016a]. Inspired by the studies, we will employ a mask-based generalized eigenvalue beamformer (GEV) with direct mask prediction.

Apart from front-end approaches that try to improve robustness by explicitly removing reverberation, we also focus on the backend. It is implemented by Probabilistic Linear Discriminant Analysis (PLDA, Section 2.4.2) on top of i-vector embeddings (Section 2.3.1) in our SV system. In order to adapt it to a new channel, introduced by either room conditions or pre-processing, we designed multiple training datasets to capture these channel effects. We eventually compare their impact on the system performance.

This work was summarized in our paper presented at Interspeech 2018 [Mošner et al., 2018b]. It is a continuation of our initial experiments in [Mošner et al., 2018a], where we analyzed the aforementioned methods on a limited amount of data. This work focuses on system robustness and consistency across multiple environments, as we perform tests on data obtained from three different rooms. In addition to the analysis presented in [Mošner et al., 2018a], we also used real and simulated multi-channel data in the PLDA training to improve adaptation to the test scenario. We also examined the correlation of the SV system accuracy with D50[1] (definition) acoustic parameter [Kuttruff, 2009] and explored a possible scenario for SV when we select the best possible single channel and avoid the multi-channel processing.

### 6.3.2 Speaker Verification System Description

We used a speaker verification system considered a gold standard when conducting our experiments. It used Mel-frequency cepstral coefficients (MFCCs) and i-vector representations of recordings [Dehak et al., 2011]. 19 MFCC coefficients were extracted along with log-energy from 25 ms windows with 10 ms overlaps of the input audio. MFCCs were augmented with their $\Delta$ and $\Delta\Delta$ coefficients, providing 60-dimensional feature vectors further subjected to the cepstral mean and variance normalization (CMVN) with a 3-second sliding window. A gender-independent 2048-component GMM-UBM was trained on a subset of the PRISM dataset [Ferrer et al., 2011] that consists of 15,600 telephone and microphone recordings that include both female (1,174) and male (813) speakers. We employed an i-vector extractor, where the dimensionality of the i-vector space was 600. The i-vectors

[1] D50 aims to quantify speech clarity and intelligibility in reverberant enclosures.

were then projected to a 200-dimensional space employing Linear Discriminant Analysis (LDA). Finally, we used PLDA as our backend. The dimensionality of latent variables in PLDA was 200 as well.

### 6.3.3 Experimental Setup

#### WPE and Dereverberation/Denoising Autoencoder

Weighted Prediction Error (WPE) is a statistical dereverberation method that tackles late reverberation via delayed linear prediction in the short-time Fourier transform (STFT) domain. Specifically, the current desired speech signal (assumed to be the sum of the direct signal and early reverberation) is estimated using regression coefficients applied to the past frames with a specific delay (i.e., not considering frames immediately following the current one). The method assumes that the desired signal is a realization of a Gaussian process where the variance is time-varying. Under the model, parameters (regression coefficients and a set of variances) are estimated via maximization of the joint likelihood of the observed signal frames. The approach is blind, requiring no prior knowledge of the room impulse response, and has been widely adopted as an effective pre-processing technique for far-field applications. In order to perform the WPE-based dereverberation, we divided audio signals into frames of 32 ms by shifting the Hann window by 8 ms. The number of regression coefficients was set to 15.

The autoencoder used for dereverberation/denoising consists of three hidden layers, each comprising 1,500 neurons. A central frame of a log-magnitude spectrum with a context of $\pm 15$ frames forms the input to the autoencoder. Taking into account the context, the dimensionality of the input vectors is 3,999. A 129-dimensional enhanced central frame represents the output. The network was trained by optimizing the mean squared error (MSE) between the output and the clean reference. The training corpus comprised the Fisher English database parts 1 and 2, yielding approximately 1,800 hours of audio. The datasets were artificially corrupted with noises from the Freesound[2] library at the SNR levels from 0 to 21 dB. RIRs for reverberation were taken from the AIR database [Jeub et al., 2009]. More details on the autoencoder and the training procedure can be found in [Plchot et al., 2016]. The network will be referred to as *DNS* in this text.

#### Beamforming Techniques

In order to perform beamforming, we used two different methods as outlined in Section 6.3.1. The first one is a basic delay-and-sum (*DS*) beamformer, the second one is the GEV beamformer, which uses a neural network to directly predict masks (see Figure 5.4a) for mask-based beamforming.

For the DS beamforming, the input signals are divided into 500 ms overlapping frames multiplied by the Hann window. The time difference of arrival (TDOA) is determined for every frame, where we assume one reference channel. Then, the delays are estimated with respect to it.

For the GEV beamformer mask estimation, we used a feed-forward neural network with the architecture that was defined in [Heymann et al., 2016a]. We explored two training schemes. In both, we used 7,138 recordings of the simulated training data defined by the 3rd CHiME challenge [Barker et al., 2015]. In the first scheme, the data was prepared[3]

[2] https://freesound.org/
[3] https://github.com/fgnt/nn-gev

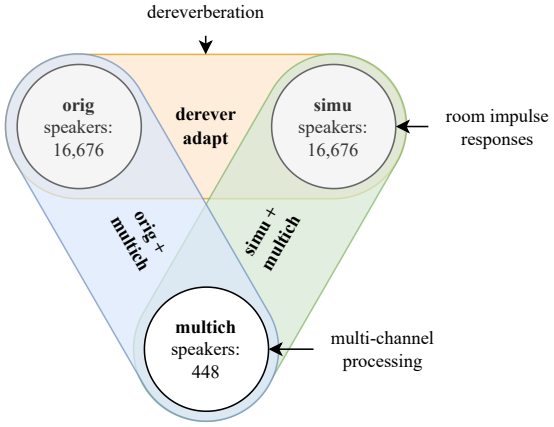



Figure 6.2: Illustration of the datasets used to re-train the PLDA backend. The set termed *multich* is never used in isolation.

according to [Heymann et al., 2016a]. The beamformer using this neural network will be referred to as *FW_GEV*. For the second scheme, we prepared another dataset that specifically targeted dereverberation. It was obtained by convolving the original training recordings with parts of random room impulse responses (RIRs). Following [Heymann et al., 2017], we considered a convolution of the speech signal with the first 50 ms of the RIR to represent clean audio. Convolution with the rest of the RIR resulted in a signal used for residual mask computation. The beamformer using the mask estimator trained on such data will be denoted as *FW_GEV_rever*.

### PLDA Training Data Augmentation

Glembek et al. concluded that the PLDA is a key part to focus on when adapting the i-vector-PLDA SV system [Glembek et al., 2014]. Following this conclusion, we retrained our PLDA model on different datasets of specific properties. Figure 6.2 provides a pictorial representation of the designed datasets for better understanding. In order to focus on channel variability, all datasets contain the same set of 16,676 speakers.

**orig** The original training data comprised 86,629 telephone and microphone recordings from the PRISM corpora [Ferrer et al., 2011], covering 9,663 female and 7,013 male speakers. The system trained on this data is considered our baseline. We note that the i-vector extractor was also trained on this data.

**simu** The dataset was designed to resemble the test data regarding the acoustic channel. It contains the same 86,629 recordings as the *orig* dataset. However, all of them were convolved with simulated room impulse responses. We used the image source method [Allen and Berkley, 1979] to obtain RIRs. Room dimensions were chosen randomly for every recording. Minimum room size was set to $2.0 \times 4.0 \times 2.3\,\mathrm{m}$, maximum to $10.0 \times 12.0 \times 5.0\,\mathrm{m}$. The placement of the source and the receiver was randomized as well.

**orig + multich** We selected data from previous NIST Speaker Recognition Evaluations (2005, 2006, 2008, 2010) that were simultaneously recorded with multiple (4 to 14) microphones. We picked 3,490 utterances from 448 speakers already present in the *orig*

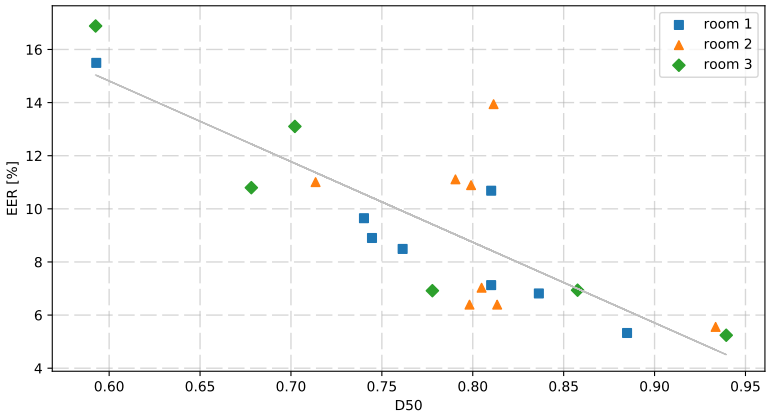



Figure 6.3: Distribution of EERs versus the D50 values corresponding to individual microphones in the test rooms.

dataset. Before the PLDA training, the multi-channel data was processed using the same pipeline used to process the evaluation data (see Section 6.3.4). The final dataset combines the resulting single-channel recordings and those from the *orig* dataset, yielding 90,119 examples.

**simu + multich** This dataset is almost identical to *orig + multich*. The only difference is that the *orig* data was replaced by *simu*.

**derever adapt** The base for *derever adapt* is *orig + simu*. The simulated part was further processed using the corresponding dereverberation method.

### 6.3.4 Evaluation of Dereverberation and Backend Adaptation

In order to evaluate our systems, we employed speaker verification evaluation data of ReverbDB described in Section 4.2. Results of our experiments are presented in Table 6.1 in terms of equal error rate (EER) computed on pooled scores from all three presented ReverbDB rooms. The columns correspond to systems (training datasets). The table is divided into row sections according to the test data pre-processing. The (*clean*) evaluation data corresponds to source speech used for retransmission as described in Section 4.2. Such recordings were used to show the potential top-line performance of the SV with the given architecture. The clean evaluation data were then used during the retransmission process in the three rooms to yield data denoted as *reverberant*. The top-line EER of 2.71% was obtained by evaluating the clean test data using the *orig* system. This work aims to approach such accuracy with reverberant evaluation data. We did not apply pre-processing to the evaluation data in the rows denoted as *reverberant*. In those rows, a substantial deterioration of accuracy is apparent. We show only the results attained with the best and the worst performing single microphones for convenience.

#### Reverberant Test Data and Dereverberation

It was shown that the definition (D50) acoustic parameter [Kuttruff, 2009] closely correlates with speaker recognition performance [Peer et al., 2008]. Inspired by such an observation,

we show the EER values as a function of D50 for all microphones in all considered rooms in Figure 6.3. In order to get the EER values, we used the baseline SV system as described in Section 6.3.2 (i.e., using the *orig* training data). Equal error rates vary substantially across rooms, and the correlation with D50 approximately holds. The value of the conducted analysis is as follows. Knowing the D50 values beforehand can be beneficial for selecting the best microphone and using a simple single-channel speaker verification. The accuracy attained with the microphones chosen according to the best and the worst (the highest and the lowest) D50 values is shown in brackets in the row with results on reverberant data in Table 6.1 (section *basic*). The EERs corresponding to the global minimum and maximum over all microphones are also in the table (values without brackets, section *basic*). The results confirm that the choice according to D50 is reasonable, as the numbers with and without brackets match with only one exception.

From the *dereverb.* section, it is evident that testing on dereverberated data captured by the best-performing microphone leads to improved performance. When the WPE dereverberation is applied, the reduction in EER becomes more pronounced. Consistently, the microphone delivering the highest accuracy in most cases corresponds to the one selected based on the highest D50 value. Notably, the best EER achieved using WPE (2.62%) even surpasses that obtained on the original test data (2.71% EER). Although the selection using D50 seems appealing, it is essential to emphasize that using D50 values assumes ideal conditions. These values are computed from room impulse responses (RIRs) and depend on the positions of the speaker and microphone, which must be known in advance. In practical applications, such information and the acoustic properties of the target environment are often unavailable. Moreover, as shown in the results table, the microphones with the lowest performance do not necessarily correspond to those with the lowest D50 values. In some acoustic conditions, dereverberation can even degrade performance, altering which microphone is the least effective. Our experiments demonstrate that the neural network-based dereverberation method tends to produce more consistent and stable results, as results with the worst-performing microphone improve compared to no pre-processing (which may not be the case for WPE).

### Pre-Processing With Beamforming

We compare two approaches that are rather different in complexity but may work blindly, requiring neither speaker nor microphone locations. The results obtained with beamformed evaluation data with differently adapted SV systems are shown in the *beamf.* section of Table 6.1.

The results suggest that the *orig* system performance can be enhanced by augmenting the PLDA training data with examples processed by the multi-channel method applied also to the evaluation data (*orig + multich*). However, providing the PLDA with recordings convolved with RIRs during training seems more beneficial despite a smaller number of recordings (*simu*). A possible explanation is that beamforming performs a combination of linearly filtered reverberant data. The channel of such data could match the linearly filtered training data (using RIRs) better than the channel of the original data. Further improvement is achieved by introducing beamformed real multi-channel data to the reverberant dataset (*simu + multich*).

Table 6.1: Effect of pre-processing and PLDA adaptation. BUT ReverbDB data was used for evaluation. Scores from the three rooms were pooled to compute EER [%]. Values in brackets were selected according to the best/worst values of D50.

| | Test data | | orig | simu | orig + multich | simu + multich | derever adapt |
|---|---|---|---|---|---|---|---|
| basic | clean | | 2.71 | 2.73 | - | - | - |
| | reverberant | best | 5.24 (5.24) | 4.31 (4.59) | - | - | - |
| | | worst | 16.88 (16.88) | 12.37 (12.37) | - | - | - |
| dereverb. | DNS | best | 3.90 (3.90) | 4.30 (4.30) | - | - | 3.64 (3.64) |
| | | worst | 12.57 (10.17) | 12.66 (10.82) | - | - | 9.12 (7.97) |
| | WPE | best | 2.73 (2.73) | 3.67 (3.67) | - | - | 2.62 (2.93) |
| | | worst | 19.71 (9.12) | 18.34 (8.60) | - | - | 17.49 (8.48) |
| beamf. | DS | | 7.94 | 6.32 | 7.66 | **6.01** | - |
| | FW_GEV | | 5.84 | 5.07 | 5.49 | **4.75** | - |
| | FW_GEV_rever | | 5.98 | 4.89 | 5.49 | **4.82** | - |
| DNS + beamf. | DS + DNS | | 6.96 | 7.86 | 6.15 | 6.64 | **5.45** |
| | FW_GEV + DNS | | 5.97 | 6.01 | 5.00 | 5.28 | **4.51** |
| | FW_GEV_rever + DNS | | 5.87 | 5.70 | 4.82 | 4.96 | **4.22** |
| WPE + beamf. | WPE + DS | | 5.10 | 5.91 | 4.89 | 5.14 | **4.68** |
| | WPE + FW_GEV | | 3.32 | 3.74 | 3.21 | 3.59 | **3.07** |
| | WPE + FW_GEV_rever | | 3.35 | 3.95 | 3.18 | 3.39 | **3.15** |
| combined | WPE + DS + DNS | | 5.74 | 7.79 | **5.00** | 6.01 | - |
| | WPE + FW_GEV + DNS | | 3.63 | 4.47 | **3.46** | 3.74 | - |
| | WPE + FW_GEV_rever + DNS | | 3.66 | 4.37 | **3.11** | 3.60 | - |

### Combination of Beamforming and Dereverberation

In this part, we explore the effects of combining beamforming and dereverberation methods. According to our experiments, the order in which the methods are applied is important.

Regarding the dereverberation/denoising autoencoder, we observed that neural network processing should be preceded by beamforming. When swapping the order, the DNS is applied separately to individual channels. It performs highly non-linear operations; therefore, it breaks the assumptions of the beamforming about the input channels. Moreover, it likely corrupts phase information. The corruption is problematic for DS, which relies on delay estimates provided by the GCC-PHAT. Mask estimation in FW_GEV also performs poorly when confronted with DNS-processed recordings, as it was trained on data from a different domain. The role of the mask estimator then also becomes ill-defined.

Focusing on the *DNS + beamf.* section of Table 6.1, we observe that appending DNS to beamforming led to performance improvement in two out of three cases when using PLDA trained on the *orig* data compared to beamforming-only pre-processing. On the other hand, when employing the *simu* PLDA, integrating DNS results in deterioration, likely due to increasing mismatch between training and test data. Indeed, introducing the training data processed by beamforming and subsequent DNS brings improvements (*orig*|*simu + multich*). Since *orig + multich* works promisingly, we trained another system, making use of simulation to produce more training data that is subsequently dereverberated by DNS (*derever adapt*). This resulted in a further average improvement.

We also experimented with WPE instead of DNS. In that case, WPE should be applied before beamforming. Since WPE can process multi-channel signals and output multi-channel signals, it is perfectly suitable for this scheme[4]. The *simu* system yields the worst results among the systems shown in the *WPE + beamf.* section of Table 6.1. It corresponds

[4]We also observed that multi-channel WPE is more efficient than the single-channel alternative.

to the ability of the WPE to dereverberate signals. In the previous experiments, we observed that augmenting the PLDA training data with multi-channel real data processed by the same pipeline applied to the evaluation data improves performance. When such data, originating from real conditions, was added to *simu*, creating *simu + multich*, a considerable improvement in EER was observed. Despite the gain, *simu + multich* still performs worse than the *orig* system. As expected, the performance of the *orig* system augmented with multi-channel data processed consistently with the test data (*orig + multich*) improved over the *orig*. Finally, we performed experiments using the *derever adapt* training data. It provided the best-performing system for the given pre-processing pipeline. This outcome is consistent with that for the pipeline comprising beamforming and the DNS. Therefore, we can draw an important outcome of the experiments: Acquisition of a sufficient amount of genuine multi-channel labeled data is not straightforward (demonstrated by *orig + multich*). However, one can make use of simulated data. Despite the lack of realism stemming from the image source method, data can be easily generated in great volumes. Then, one can obtain a system with performance comparable to a system trained using real data. However, we should stress that this conclusion holds for the specific setting and scenario under investigation.

Intuitively, applying all of the pre-processing steps is tempting due to the imperfections of the individual techniques. We first applied WPE, beamforming, and finally DNS according to the ordering of methods determined before. The results in the *combined* section of Table 6.1 show that no improvement over *WPE + beamf.* is achieved. The compound pre-processing pipeline tends to provide results lying between *DNS + beamf.* and *WPE + beamf.*

### 6.3.5 Conclusions of the Experiments

We analyzed multiple techniques to deal with the far-field multi-channel speaker verification. We aimed to develop a robust system that can perform consistently in various acoustic conditions. In the data pre-processing phase, we explored the effects of WPE and autoencoder dereverberation. Moreover, we applied blind beamforming (delay-and-sum and generalized eigenvalue beamformer) to multi-channel data from ad-hoc microphone arrays. In order to adapt the SV system, we used both simulated and real reverberant data.

We observed that applying the WPE and then choosing an appropriate microphone can deliver approximately the same accuracy as the top-line (using clean evaluation data). However, relying on selecting the best microphone is questionable, as prior knowledge about the settings of the actual room is rarely available, and the existence of such a good microphone is not guaranteed.

Overall, the best results were achieved when *WPE + beamf.* was used for the evaluation data pre-processing and either *orig + multich* or *derever adapt* systems were used for verification (see Table 6.1). Out of the beamforming techniques, FW_GEV was the best in this setting. FW_GEV_rever yielded somewhat worse results, but they were still close to the best ones. Regarding EER, the best combination yielded 3.07% EER, which is a 13.5% relatively worse result compared to the accuracy on the clean test data. However, it also means 41.4% relative improvement over the best possible microphone, showcasing the robustness of the designed SV system, benefiting from multi-channel data pre-processing and careful training data augmentation.

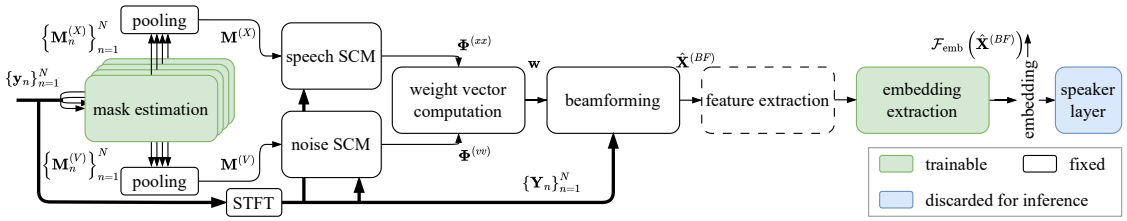



Figure 6.4: Block diagram of the proposed multi-channel embedding extraction framework.

## 6.4 Multi-Channel Speaker Verification via Cascaded Beamforming and Speaker Embedding Extraction

In Section 6.3, we were concerned with multi-channel speaker verification in the presence of reverberation. In our experiments, we designed a system combining multi-channel preprocessing and the PLDA adaptation (with carefully selected data) that nearly matched the performance of a clean-domain system of the same backbone. These results suggest that the problem of reverberant data becomes less of an issue when employing microphone arrays. Therefore, we will further aim to tackle more realistic and challenging scenarios in the rest of the text. When speech is recorded with distant microphones, not only does reverberation corrupt it. Other interfering sources likely become active. By distractors, we mean television, radio, music, HVAC (heating, ventilation, and air conditioning), and domestic sounds (to name a few). We generally call them noises.

In this section, we will describe a general framework that we gradually built in multiple studies. We will start by presenting our published work and will finish by providing more results and insights.

### 6.4.1 General Framework

In general, our framework, shown in Figure 6.4, comprises two logical components: 1) multi-channel processing that provides a single-channel output containing enhanced speech for a downstream model, and 2) speaker-descriptive vector (i.e., embedding) extraction. Despite the complexity of the design we propose, it has multiple benefits over structurally simpler models directly predicting speaker embeddings given multi-channel audio signals (i.e., models with late fusion and model-level approaches described in Section 6.2):

1. Both front-end and embedding extraction models can be initially developed independently, reflecting advances in the respective research fields. Consequently, the specific models we will present in this work are just representatives of the entire family of solutions. They showcase recent research directions.

2. Our framework allows for a better understanding of model predictions. Multiple human-interpretable intermediate results are produced when processing data, such as an enhanced signal after the pre-processing stage or beamforming weights that can be used to obtain the beampattern [McDonough and Kumatani, 2012]. Such clues can shed light on the decisions the model makes.

3. Nowadays, home devices use microphone arrays where the position of microphones is fixed and known in advance, and their properties can be described by theory. We deal with a more general scenario: microphone arrays comprise sensors distributed

around rooms at unknown locations. Moreover, our solution is agnostic to the number of channels by design. Therefore, the resulting models are versatile and not tailored towards specific microphone array configurations.

4. Our solution was designed with the scarcity of multi-channel data for speaker verification in mind [Mošner et al., 2022b]. Current state-of-the-art embedding extractors, such as versions of ResNets [Chen et al., 2023, Huh et al., 2023] or models based on self-supervised learning (SSL) networks [Peng et al., 2023a, Chen et al., 2022b], are data-demanding. It is reasonable to expect that the amount of data required by multi-channel embedding extractors would be on a similar scale. The decomposition of the problem to multi-channel pre-processing and single-channel embedding extraction allows us to utilize plentiful data (such as VoxCeleb 1,2 [Nagrani et al., 2017, Chung et al., 2018] or VoxBlink 2 [Lin et al., 2024a]) for the latter. Advantageously, as evidenced by [Hershey et al., 2016, Cosentino et al., 2020], powerful source separation models are less data-demanding. It renders our speech-noise MultiSV corpus usable for the front-end (see Section 4.3 and [Mošner et al., 2022b]).

Our framework's core of multi-channel processing is linear spatial filtering — beamforming. Recently, variants of all-neural beamformer models have been published [Zhang et al., 2021b, Zhang et al., 2021a, Xu et al., 2021, Gu et al., 2023]. Despite their promising performance, we stick to well-established beamformers described in Section 3.2. We propose a general framework and provide a retrospective analysis w.r.t. (not only) pre-processing. Our study has by no means a chance to cover all the directions in separation/enhancement and SV. Therefore, variations (including all-neural beamforming) have not been considered due to the unbearable breadth of the required analysis.

The second stage, which is entirely single-channel, is specific to the target application. Although we focus on predicting speaker embeddings, the second-stage model can be arbitrary (for instance, an acoustic model for ASR, a language identification system, etc.).

In Figure 6.4, we also show a feature extraction standing between the two main components. It is optional and depends on the input required by the extractor. Since the beamformer output is in the short-time Fourier transform (STFT) domain, at least its inversion is needed (for time domain extractors [Peng et al., 2023a]).

### Formal Description

In pre-processing, we aim to create a mapping function $\mathcal{F}_{\text{enh}} : \mathbb{C}^{T\times F\times N} \to \mathbb{C}^{T\times F}$; $\mathbf{Y} \mapsto \hat{\mathbf{X}}^{(BF)}$ that estimates a speech image at the reference microphone. Denoting the embedding extractor (with optional feature extraction) as $\mathcal{F}_{\text{emb}} : \mathbb{C}^{T\times F} \to \mathbb{R}^I$, with $I$ denoting the embedding dimensionality, the whole framework is given as $\mathcal{F}_{\text{emb}} \circ \mathcal{F}_{\text{enh}} : \mathbb{C}^{T\times F\times N} \to \mathbb{R}^I$.

As beamforming is performed following the broadband scheme presented in Section 3.3, the enhanced single-channel output $\hat{\mathbf{X}}^{(BF)}$ is in the STFT representation. When necessary for compatibility reasons, it is transformed to log Mel-filterbank energy (*fbank*) feature vectors with a different frame rate (through the time domain).

The embedding extractor maps its input to per-frame representations $\mathcal{F}_{\text{per_frame}}(\hat{\mathbf{X}}^{(BF)})$ that are subsequently aggregated over time via statistics (mean and standard deviation) pooling [Snyder et al., 2017]. The resulting vector is transformed into the embedding space with the transformation matrix $\mathbf{E} \in \mathbb{R}^{P\times I}$ with $P$ representing the pooled vector space's

dimensionality. The embedding extractor can be summarized as

$$\mathcal{F}_{\text{emb}}\left(\hat{\mathbf{X}}^{(BF)}\right) = \Big[\boldsymbol{\mu}_{\mathcal{F}_{\text{per_frame}}(\hat{\mathbf{X}}^{(BF)})}; \boldsymbol{\sigma}_{\mathcal{F}_{\text{per_frame}}(\hat{\mathbf{X}}^{(BF)})}\Big]\mathbf{E} \in \mathbb{R}^{I}, \tag{6.3}$$

where $\boldsymbol{\mu}_{(\cdot)}$ and $\boldsymbol{\sigma}_{(\cdot)}$ stand for per-dimension mean and standard deviation operations, respectively. The operator $[\cdot;\cdot]$ concatenates its arguments.

### 6.4.2 Outline of the Framework Development

In the following text, we will focus on different parts of the framework: 1) speaker-noise separation, 2) beamforming, and 3) embedding extraction and verification score computation. They will be covered in the following sections:

- **Section 6.4.3** will present a proof-of-concept system that we designed and presented in [Mošner et al., 2019]. 1) Speech separation was based on direct mask prediction following Section 5.5.1. 2) The system adopted the mask-based GEV beamformer. We will discuss the reasons for such a choice and the consequences for optimization. 3) Speaker embeddings were extracted with a neural network designed for that purpose, TDNN [Snyder et al., 2017]. In order to obtain speaker verification scores, we employed the PLDA model.

- **Section 6.4.4** will be concerned mainly with designing an indirect mask predictor (see Figure 5.4b) within the framework. The section will follow the design we published in [Mošner et al., 2022a]. 1) Speaker-noise separation was performed by the Conv-TasNet model [Luo and Mesgarani, 2019]. Since it provides time-domain outputs, the masks for beamforming were estimated indirectly. 2) Given the difficulties encountered with the GEV beamformer, it was replaced by MVDR. 3) We updated the TDNN speaker embedding extractor with a newer model, ResNet34 [He et al., 2016a]. Due to a different optimization criterion, we eventually used cosine scoring instead of the PLDA backend.

- In **Section 6.4.5**, we will concentrate on the framework analysis, focusing on the front-end and joint fine-tuning of different system components. 1) We will explore various separation models and their alternatives, described in Sections 5.5 and 5.7. 2) We will stick with the MVDR beamformer. 3) To keep the experiments tractable, we reduced the variability using a single updated ResNet34-based speaker embedding extractor and cosine scoring.

### 6.4.3 Multi-Channel Speaker Verification with Application-Aware Beamforming and PLDA

At the time of our experiments described in this section, x-vectors brought a new and very convenient scheme of discriminative training. At the same time, the Probabilistic Linear Discriminant Analysis [Prince and Elder, 2007] was still kept as a generative backend on top of x-vectors to perform the actual speaker verification task and, in the end, to provide scores for speaker verification trials. Following the outlined trend and inspired by improvements brought by x-vectors in single-channel far-field SV [Matějka et al., 2019], we adopted x-vector embeddings and the PLDA backend into our pipeline.

In our previous experiments, we observed that beamformers can process multi-channel signals such that the speaker verification performance is enhanced, despite optimizing different criteria (from speaker recognition) in the first place. Analogous conclusions were also drawn in the ASR field. Inspired by these outcomes, some works took the next step by making the beamforming aware of the target (ASR) application through joint optimization. For example, in [Li et al., 2016,Sainath et al., 2016,Sainath et al., 2017], multi-channel time-domain convolution filters that take raw audio signals and map them down to a single channel via filter-and-sum operation were prepended to an ASR DNN. By training the whole model (including an acoustic model and filter-and-sum) together, optimizing the ASR objective, the beamforming was forced to provide outputs beneficial for the speech-to-text system. Such an approach brought accuracy improvements at the expense of a significant volume of training data. Moreover, it assumed a fixed configuration of microphones. The issue of fixed microphone arrays was overcome in [Heymann et al., 2016a], which employed acoustic mask-based beamforming (GEV) using neural network estimates of masks for the ASR task. The follow-up work demonstrated that this beamforming method can also be successfully integrated with the ASR system and trained jointly to boost performance [Heymann et al., 2017].

The abovementioned ASR studies and our prior experiments with mask-based beamforming motivated us to integrate it with neural-network-based embedding extraction. In our case, we could not train both models jointly as we did not have a large enough database of multi-channel training data with speaker labels. Therefore, we kept the parameters of an existing x-vector model fixed. The exception was the last layer of that model, which we trained for a few iterations on a set of speakers in the multi-channel dataset. The mask predictor for the beamformer plugged into the pipeline was either randomly initialized or pre-trained and then fine-tuned in a target application-aware fashion. For the front-end pre-training, we experimented with two strategies: 1) optimizing the mean squared error (MSE) between the source and beamformed signals (please note that this does not correspond to optimizing the direct mask predictor using MSE between mask outputs and IRMs explored in Section 5.6.1), and 2) optimizing the binary cross-entropy (BCE) between the values in the estimated spectral masks and precomputed IBMs. Fine-tuning concerning the target application involved training the mask estimator with the cross-entropy objective using the x-vector model outputs.

**Beamforming — Multi-Channel Pre-Processing**

In the following experiments, we chose the mask-based GEV beamformer in (3.49)[5]. The selection was motivated by multiple reasons:

- It brought considerable performance gains for ASR in the 4th CHiME challenge [Heymann et al., 2016b], and it was successfully combined with an acoustic model and trained jointly [Heymann et al., 2017].
- The beamforming weights can be estimated using speech and noise spatial covariance matrices. Those are predicted with the help of a neural network. In turn, the neural network becomes an integral part that can be fine-tuned with the final speaker-discriminative objective conditioned on the differentiability of the beamforming operation.

[5]Specifically, we used a broadband version of the GEV following the description in Section 3.3.

- The approach with neural network-based mask estimation makes beamforming agnostic to the number of microphones. Internally, it is achieved by estimating per-channel masks with the same predictor.

### Component Connection and Gradient Propagation

The mask estimator and embedding extractor need to be connected such that error back-propagation to the beginning of the network is allowed. In our scenario, we assume the application-specific model has been trained and is differentiable to allow back-propagation to the beamforming component. Only the parameters of the mask estimator will be updated while keeping the x-vector extractor weights fixed. The beamformer outputs frames of a complex spectrum, while the x-vector extractor we use takes Mel-frequency cepstral coefficients (MFCC) as inputs. The issue with the downstream model is that its frame rate may differ from that of a beamformer, which is the case in our model. Even though, according to [Heymann et al., 2017], changing the mask estimator frame rate to match the frame rate of an acoustic model has no significant impact on speech recognition results, we opted for a more principled approach since our analysis on frame-rate revealed the importance of a frame size (Sections 5.4 and 5.6.1). Therefore, we perform the following steps to allow different frame rates of the models: Inverse STFT is applied to the beamformer output to obtain a time domain signal, which is then transformed back to the STFT domain with a different frame size and shift. These operations are differentiable, hence the error back-propagation is not disrupted. The described operations are encapsulated in the *feature extraction* in Figure 6.4.

As shown in Figure 6.5, the eigenvalue decomposition required by GEV stands between the mask estimator and the beamformer. In order to update weights of the mask estimator using error propagation from the x-vector extractor, differentiability of the eigenvalue decomposition w.r.t. its input is required. It is not a straightforward task [Boeddeker et al., 2017]. At the time of our experimentation, widespread neural network libraries supported differentiable eigenvalue solvers (where the input matrix is Hermitian) but not the generalized ones.

We approached the generalized eigenvalue problem solution by turning it into the eigenvalue problem via Cholesky decomposition of the noise SCM: $\mathbf{\Phi}_f^{(vv)} = \mathbf{L}_f \mathbf{L}_f^{\mathsf{H}}$, where $\mathbf{L}_f$ is a lower triangular matrix. The problem defined in (3.50) is then reformulated as follows:

$$\begin{aligned} \mathbf{\Phi}_f^{(xx)} \mathbf{w}_f^{(\mathrm{GEV})} &= \lambda \mathbf{L}_f \mathbf{L}_f^{\mathsf{H}} \mathbf{w}_f^{(\mathrm{GEV})} \\ [\mathbf{L}_f^{-1} \mathbf{\Phi}_f^{(xx)} (\mathbf{L}_f^{\mathsf{H}})^{-1}] \mathbf{L}_f^{\mathsf{H}} \mathbf{w}_f^{(\mathrm{GEV})} &= \lambda \mathbf{L}_f^{\mathsf{H}} \mathbf{w}_f^{(\mathrm{GEV})} \\ \mathbf{A}_f \mathbf{y}_f &= \lambda \mathbf{y}_f. \end{aligned} \tag{6.4}$$

The last line represents the eigenvalue problem with Hermitian matrix $\mathbf{A}_f$ and thus may be solved using available routines that support gradient propagation. However, gradient propagation through the Cholesky decomposition we introduced is needed. It was available in some libraries (such as TensorFlow) during our experimentation. As follows from (6.4), beamforming weights are obtained by a linear transformation of the eigenvector corresponding to the highest eigenvalue: $\mathbf{w}_f^{(\mathrm{GEV})} = (\mathbf{L}_f^{\mathsf{H}})^{-1} \mathbf{y}_f$. At this point, the generalized eigenvalue problem is differentiable, and errors from the x-vector extractor can be used to update the mask predictor parameters.

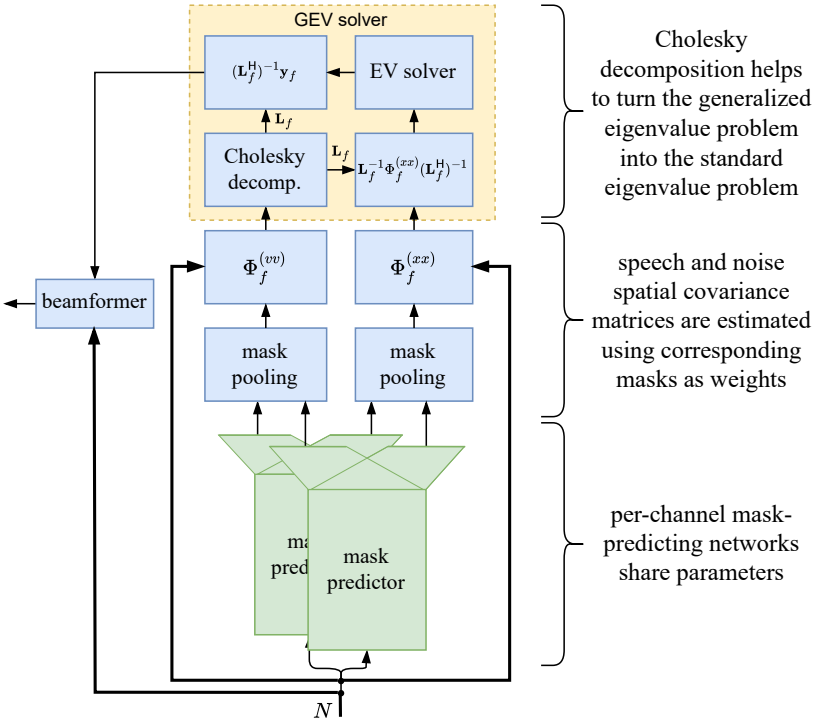



Figure 6.5: Detailed mask-based GEV beamformer, realized through Cholesky decomposition, with an integrated neural network. Green blocks represent trainable parts; blue ones are fixed. The computation flow is displayed for one frequency bin. $N$ stands for the number of channels (4 in our case). $\mathbf{\Phi}_f^{(vv)}$ and $\mathbf{\Phi}_f^{(xx)}$ represent noise and speech SCMs, respectively. $\mathbf{\Phi}_f^{(vv)}$ is decomposed employing Cholesky decomposition. The eigenvalue (EV) solver computes a principal eigenvector $\mathbf{y}_f$ of the input matrix. The transformed $\mathbf{y}_f$ corresponds to the principal eigenvector of the generalized eigenvalue (GEV) problem and is used as the weight vector for beamforming.

### Fine-Tuning Data

Since the MultiSV dataset was not established when we conducted these experiments, we needed to prepare training, development, and evaluation data. We turned the definition of development and evaluation sets for the VOiCES challenge [Nandwana et al., 2019] into multi-channel trials. Therefore, they are equal to the development and evaluation protocols of *MultiSV SRE*, respectively. Our training corpus comprised a set of recordings from room 1 and room 2[6] (see Figures 4.2a and 4.2b). Since the development data was also recorded in these rooms, we needed to filter it out from the training set. We also removed files where we found a discrepancy between the lengths of source LibriSpeech and retransmitted recordings, which would cause problems in the mask estimator pretraining. In the following step, groups of four recordings were merged by speaker identity,

[6] Data from room 3 and room 4 were not released back then.

Table 6.2: Performance of the x-vector extractor that serves as an application-specific model. Prior target probability $P_{\text{tgt}}$ for the minimum detection cost minDCF was set to 0.01.

| | **EER [%]** | **minDCF** |
|---|---|---|
| VOiCES dev | 2.03 | 0.261 |
| VOiCES eval | 5.51 | 0.459 |

chapter, segment, room, and distractor type[7], ensuring that all four microphones recorded the same session. Microphones in the microphone arrays were chosen randomly to enhance the diversity of the set. The resulting training dataset consisted of 57,800 examples (arrays comprising four microphones) spanning the voices of 200 speakers. The average length of utterances was approximately 15 seconds.

### X-Vector Extractor Architecture and Re-Training

As the application-specific model, we use the x-vector extractor corresponding to model 14 in Table 2 of [Matějka et al., 2019]. Its architecture was presented in Table 2.1. The model was trained with the Kaldi toolkit [Povey et al., 2011] following the `SRE16/v2` recipe with several modifications [Matějka et al., 2019]. The training data comprised 6.2 million speech segments from 7,146 VoxCeleb 1 and 2 dev set speakers. Out of them, 1.2 million corresponded to the original audio. The remaining 5 million segments were augmented versions. All training segments were 200 frames long, and the model was trained for nine epochs. In order to put its performance into context, we show results on the single-channel VOiCES development and evaluation sets in Table 6.2.

The x-vector extractor network is trained to classify speakers in the training set (VoxCeleb in our case). Eventually, we require the classifier error to be propagated to the mask estimator. However, the VOiCES multi-channel (dataset used for the speech enhancer training) comprises different speakers. In order to enable the x-vector extractor to classify the 200 VOiCES speakers, we replaced the final linear layer and retrained it while keeping the rest of the parameters frozen. Since x-vectors are extracted from one of the preceding layers, this modification does not affect them (considering one specific input to the network).

### Speaker Verification Backend

We used the PLDA backend identical to that in the Kaldi x-vector recipe. This backend involves a pre-processing step which first reduces the x-vector dimension by LDA from 512 to 250[8], and then applies a nonstandard variant of length-normalization[9]. We concatenated all segments from each VoxCeleb 1 and 2 dev session (having the same YouTube identifier) to train the backend. Including augmentations, this resulted in 830 thousand files.

[7] The speaker, chapter, and segment labels were provided by LibriSpeech metadata. The room and distractor types were introduced in the VOiCES corpus.

[8] Speakers are considered as classes for the LDA estimation.

[9] https://github.com/kaldi-asr/kaldi/blob/master/src/ivector/plda.cc

### Baseline

In order to assess our proposed framework, we employ the BeamfomIt toolkit [Anguera et al., 2007] as a baseline multi-channel beamformer. It does not fit our framework definition because it has no learnable parameters, but it is well-established in the speech community. Speaker verification performance obtained with this fixed beamformer is displayed in the first row of Table 6.3. The results are shown in terms of equal error rate (EER [%]) and minimum detection cost (minDCF) as defined for the VOiCES challenge in [Nandwana et al., 2019] (i.e., the prior target probability $P_{\text{tgt}}$ is set to 0.01).

### Isolated Mask Estimator Training

In order to evaluate the effect of beamformer training in the application-aware framework, we first train the mask estimator separately and plug it into the processing chain. The resulting models of independent training will then also be used as seed models and trained further with a speaker-discriminative objective.

The mask predictor follows the direct estimation scheme in Figure 5.4a and its architecture corresponds to that described in Section 5.5.1. Dimensionality of both inputs and outputs is 513, corresponding to frames of 64 ms following [Heymann et al., 2016a]. As already outlined, we explored two approaches to mask estimator training. The first one optimized masks: the network was trained to minimize binary cross-entropy (BCE) between values in the estimated masks and those in IBMs. The exact speech and additive noise decomposition knowledge is unavailable for the VOiCES corpus as the data was retransmitted. The recording process introduces a convolutive distortion; therefore, one cannot easily obtain noisy components by subtracting the source clean recording from the replayed recording. Hence, we resorted to simulating the training data (only for the mask estimator). We used the source LibriSpeech recordings to obtain a dataset that is equal in size to the multi-channel training set. In order to approximate the nature of the VOiCES corpus, room simulation (employing the image source method) was performed, and noise was added. Reverberation time RT60 of simulated rooms was randomly drawn from the interval [0.3, 0.9] seconds. We randomly placed sources of speech and noise, and four omnidirectional microphones in each. Noises, selected from the Freesound library[10], contain shop, crowd, library, office, real fan, and street sounds. They were mixed with the source speech with SNRs ranging from 3 to 20 dB. The mask-based GEV beamformer requires knowledge of speech and additive noise. In order to satisfy this requirement, we convolved the original LibriSpeech recordings with the first 50 ms of the generated room impulse responses (RIR) to yield the speech component of the audio. The noise component was obtained by convolving the source with the RIR remainder and adding reverberant noise. An additive mixture of the two corresponds to a noisy and reverberant signal recorded by an simulated microphone. We note that this approach to data generation slightly deviates from the rest of our work on decomposing speech from noise (and MultiSV), since the noise training targets contain late speech reverberation.

Since we successfully handled the problem of back-propagating through the generalized eigenvalue decomposition, we could train the mask estimator to directly optimize the ability of the beamformer to recover the clean signal at its output. For such training, the architecture comprised all the components in Figure 6.5. To avoid sparsity of gradients [Heymann et al., 2017], we employed mean pooling during training (but also later in evaluation for

[10] http://www.freesound.org

Table 6.3: Results of application-aware and application-unaware beamforming on MultiSV SRE in terms of equal error rate (EER [%]) and minimum detection cost minDCF. *Front-end obj.* stands for the objective function used for the front-end training: CE — cross-entropy (multi-class), BCE — binary cross-entropy, MSE — mean squared error. Wherever the "+" sign is used, its operands stand for the objective functions for pre-training and application-aware training in that order.

| **Method** | **Front-end obj.** | **Dev. set** | | **Eval. set** | |
|---|---|---|---|---|---|
| | | **EER** | **minDCF** | **EER** | **minDCF** |
| BeamformIt | — | 1.73 | 0.407 | 5.11 | 0.494 |
| App-aware, scratch | CE | 1.77 | 0.202 | 4.39 | 0.456 |
| Separate | MSE | 1.91 | 0.207 | 4.77 | 0.497 |
| App-aware, pretr. | MSE + CE | 1.13 | 0.146 | 4.78 | 0.496 |
| Separate | BCE | 1.53 | 0.214 | 4.26 | 0.446 |
| App-aware, pretr. | BCE + CE | 1.37 | 0.162 | 4.02 | 0.417 |

compatibility). We used MSE to compare the magnitude of the STFT output of the beamformer with the clean source speech as a loss function. This approach is more convenient since it does not depend on knowledge of additive noise (usually available only for simulated data). Therefore, we used the *retransmitted* multi-channel training dataset without any modification. On the other hand, the downside of this approach is that it neglects the phase in the loss computation despite being available.

We collectively term the two abovementioned approaches as *separate* in Table 6.3. The mask-based beamformers obtained by performing application-unaware training outperform the baseline BeamformIt in terms of EER on the evaluation set. The BCE-optimizing estimator produces better results (as measured by all metrics) than the baseline, with a significant gain in minDCF. It yields better performance compared to the MSE-optimizing approach. It may result from ignoring the phase in the MSE training. Another hypothesis concerns the fundamental difference between the two approaches. In the approach optimizing MSE, the network must learn to estimate proper noise and speech decomposition (i.e., masks) to perform beamforming, which is presumably more difficult.

### Target-Aware Mask Estimator Training

Finally, we trained the mask estimators in our framework to optimize the target objective — cross-entropy (CE) between the x-vector extractor predictions and correct speaker classes. The training may start from a random initialization of the mask estimator parameters. In such a case, the speech enhancement component must learn a reasonable combination of channels in the beamformer framework solely based on the classification error. This experiment is referred to as *app-aware, scratch*. Instead of starting from scratch, we also employed the pre-trained mask estimators (described above) as seed models for subsequent target-aware training with the CE objective. We refer to them as *app-aware, pretr* in Table 6.3.

When training the beamformer mask estimator from a random initialization, it has to deal with the difficult task of figuring out that the beamformer requires noise and speech SCM matrices (as well as the *separate, MSE* model). However, compared to training the mask estimator with the MSE objective, speaker verification loss helps it perform better. Despite its superiority over the MSE objective-based model, it still does not reach the performance of the mask estimator independently trained with the BCE objective.

An unstable behavior was observed when we used the *separate, MSE* as a seed model for the CE training, suggesting that the MSE pre-training is not ideal. On the other hand, the BCE-optimized mask estimator was a promising seed model. The separate training forced it to support real beamforming. Nevertheless, IBMs (that the model tries to predict) are handcrafted, suggesting room for improvement. That was finally achieved via training in the proposed framework.

### 6.4.4 Multi-Channel Speaker Verification With Conv-TasNet-Based Beamformer

This section will present a multi-channel SV approach keeping the same structure: beamforming utilizing neural networks and single-channel embedding extraction. Departing from direct mask prediction (from the previous section) [Mošner et al., 2019], which we now treat as our baseline, we focus on enhancing the beamforming stage via indirect mask prediction (Figure 5.4b). In order to achieve satisfactory speech enhancement and noise suppression of the beamformer, precise estimation of SCMs is required. To this end, motivated by the trends in the source separation field, we aim to utilize the speech modeling power of time-domain models (as opposed to direct mask estimators). Similarly to [Ochiai et al., 2020], we base the SCM estimation on per-channel outputs of a time-domain model from the field of source separation. We opt for Conv-TasNet (see Section 5.5.2 and [Luo and Mesgarani, 2019]) for its performance and small size, which we even decreased to obtain a small footprint network. The contributions of our study presented in [Mošner et al., 2022a] can be summarized as follows:

- Although we also used Conv-TasNet, we proposed an SCM estimation approach that is different from that used in [Ochiai et al., 2020]. We empirically found it performing consistently better in our task of SV, especially for retransmitted data and ad-hoc microphone arrays.
- We showed significant improvements on MultiSV simulated evaluation data (*eval simu*) and comparable or better performance on retransmitted data (*eval retr v1*) w.r.t. the direct mask-predictor-based model, which has 2.6 times more parameters than the proposed network. Specific numbers of parameters are given in Tables 5.2 (64 ms model) and 5.1 (Conv-TasNet small) for the baseline and the proposed models, respectively.
- We showed the efficiency of the front-end (Conv-TasNet or mask predictor) fine-tuning in a joint model by optimizing the parameters using speaker-discriminative loss. The fine-tuning brought average relative improvements of 8.0% and 6.4% for the proposed and mask-predictor-based models, respectively.

#### Method

In the first stage of front-end processing, four channels from a microphone array are independently processed by a per-channel operating Conv-TasNet small. The original model

has been proven strong in source separation, and we aim to employ its modeling power in our scenario. We utilize it to split a single-channel input mixture into corresponding speech and noise components, eventually turned into the corresponding spatial covariance matrices. Any beamformer that uses SCMs, such as MVDR or GEV, may be used. In Section 6.4.3, we showed that it is possible to make the generalized eigenvalue decomposition employed in GEV differentiable via Cholesky decomposition. Nevertheless, a standard eigenvalue decomposition is still required. It is commonly implemented using iterative procedures[11] (such as the QR algorithm). Along with the numerical instability issues, using the GEV beamforming in the end-to-end pipeline is not optimal. Therefore, we perform our analysis with the MVDR. Despite the instability caused by matrix inversion, which can still pose challenges, the MVDR tends to be more practical. After the beamforming stage, the single-channel output is subject to feature extraction and passed to a ResNet-based embedding extractor. Finally, the speaker embeddings are compared with cosine similarity (instead of PLDA) to yield SV scores.

### Conv-TasNet for Indirect Mask Prediction

In the multi-channel stage, we follow exactly the indirect mask prediction scheme for mask-based beamforming described in Section 5.6.2. We assumed that such a method could potentially neglect wrong predictions for some channels. We recall that given multi-channel mixture signals $\{\mathbf{y}_n\}_{n=1}^N$, the Conv-TasNet-based speech separator is used to obtain estimates of reverberated speech $\{\hat{\mathbf{x}}_n\}_{n=1}^N$ and estimates of noise $\{\hat{\mathbf{v}}_n\}_{n=1}^N$ for each channel independently. The time-domain signals are transformed using STFT. Given the speech and noise predictions, we employ (5.13) to obtain ratio masks. Based on our previous experiments, we combine the channel-dependent masks using product pooling.

The original best-performing Conv-TasNet model in [Luo and Mesgarani, 2019] comprises 5 M parameters. In order to obtain a more practical small-footprint model of 1.2 M parameters, called Conv-TasNet small, we altered hyperparameters as shown in Table 5.1. The rationale behind increased frame size in the encoder and decoder is to keep a similar time span of bases while using data sampled at 16 kHz (compared to 8 kHz). The original Conv-TasNet for speaker separation used the SI-SDR training objective in a PIT fashion. However, following reasoning in Section 5.5.2, we employed the ratio loss in (5.11). Moreover, it is worth stressing that scale invariance in training could cause a different dynamic range of enhanced speech and noise. As noted in [Ochiai et al., 2020], preserving scale information is important for correct subsequent SCM estimation.

### Speaker Embedding Extractor

The speaker embedding network extracts utterance-level embeddings using enhanced single-channel audio. Due to the frame size (64 ms vs. 25 ms) and shift (16 ms vs. 10 ms) mismatch between the beamforming front-end and embedding extractor, the signals are re-framed through the time domain (similarly to our previous approach).

Following the progress in the speaker embedding extraction, we replaced the TDNN-based x-vector extractor with the ResNet34 architecture [He et al., 2016a]. It follows the description in Section 2.4 (including the number of channels in stages and embedding dimensionality). Instead of 80-dimensional *fbank* feature vectors (in Table 2.2), we used 40-dimensional ones in this work. In the previous experiments with the TDNN embedding

[11] https://docs.nvidia.com/cuda/cusolver/

Table 6.4: Proof-of-concept experiments on the simulated MultiSV data.

| Front end | dev simu | | eval simu | |
|---|---|---|---|---|
| | **EER [%]** | **minDCF** | **EER [%]** | **minDCF** |
| mask predictor | 1.41 | 0.162 | 2.02 | 0.195 |
| proposed | **1.17** | **0.147** | **1.91** | **0.176** |

extractor, we used multi-class cross-entropy as the loss function. Motivated by the gains in the SV performance provided by the additive margin (AM) softmax loss (see (2.22) and [Wang et al., 2018]), we employed it as the training objective for our ResNet. We set the scale to 30. Throughout the training, the margin continuously increased from 0 to 0.2. The angular-based objective eventually affected score computation, adopting cosine similarity.

### Baseline

The baseline comprises frequency domain *direct mask predictor* as described in Section 6.4.3 in the context of our previous multi-channel SV experiments. Specifically, we selected the alternative trained using binary cross-entropy between predictions and IBMs. We emphasize that we did not use the same backend as before. For compatibility, we use the ResNet-based model instead of the TDNN-based x-vector extractor. Therefore, the results obtained with the mask predictor used here are not equal to those presented in Section 6.4.3.

### Data

We use two training datasets — one for the embedding extractor and one for the front-end. The single-channel extractor is trained on the development part of VoxCeleb 2 (see Section 4.1 and [Chung et al., 2018]) with reverberation and MUSAN noise augmentations.

Since front-end models (Conv-TasNet small and the baseline mask predictor) require speech and noise references, we use training data from the MultiSV corpus (see Section 4.3.4 and [Mošner et al., 2022b]). As front-end models process signals independently, we disregard the multi-channel nature of MultiSV. However, we fully use the multi-channel recordings in the joint training, which will be detailed later, since the beamformer uses all four channels to produce a single-channel output passed to the embedding extractor.

In order to evaluate the systems, we use trial sets defined by MultiSV. We adopt the scenario of single-channel enrollment and multi-channel test (CE and SRE). For a thorough evaluation, we employ both simulated (*simu*) and retransmitted (both *retr v1* and *retr v2*) data (see Section 4.3.3 for details).

### Proof of Concept

The first experiment aims to evaluate the ability of Conv-Tasnet small to provide good representations of speech and noise, leading to better estimation of SCMs in a controlled environment. To this end, we opted for evaluation data similar to the training data in terms of domain — both are simulated. However, differences in speakers, speaking styles (read vs. spontaneous speech), utterances, and noises make the evaluation data sufficiently dissimilar and challenging. Per results in Table 6.4, we observe consistent improvements

Table 6.5: Results on trials with retransmitted multi-channel test segments of MultiSV. Enrollment utterances are retransmitted (SRE) or clean (CE). *Sep.* means that the front-end and the speaker embedding extractor were trained independently. *Joint* training type signifies front-end fine-tuning in the joint model with the speaker discriminative loss. Frame size of STFT for beamforming was 64 ms.

| Front end | train type | params | dev retr SRE | | eval retr v1 SRE | | eval retr v2 SRE | | dev retr CE | | eval retr CE | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | EER | minDCF | EER | minDCF | EER | minDCF | EER | minDCF | EER | minDCF |
| mask pred. | sep. | 3.2M | 0.98 | 0.124 | 4.47 | 0.354 | 2.04 | 0.197 | 0.92 | 0.111 | 4.25 | 0.326 |
| proposed | sep. | 1.2M | 0.97 | 0.119 | 4.39 | 0.346 | 2.08 | 0.188 | 0.90 | 0.104 | 4.24 | 0.330 |
| mask pred. | joint | 3.2M | 0.86 | 0.109 | 4.38 | 0.329 | 2.22 | 0.182 | 0.84 | 0.087 | 4.32 | 0.315 |
| proposed | joint | 1.2M | **0.82** | **0.108** | **4.26** | **0.324** | **1.98** | **0.176** | **0.80** | **0.085** | **4.20** | **0.314** |

in all metrics with the proposed front-end over the baseline. Relative improvements range from 5.4% to 21.1%.

### Retransmitted Evaluation Data

We also provide results obtained on more realistic retransmitted data. Such an evaluation is often missing, but we find it important. The experiments we conducted assessed the ability to generalize to unseen acoustic conditions and background noises. Television and babble noises are new to the system (although they share some properties with music noise in the training data to some extent). What also makes the evaluation data difficult is the fact that the assumptions that hold for simulated training data do not need to hold for retransmitted evaluation data. Additivity of sources (speech and noise) and ray acoustics are just examples of these assumptions.

According to the results in Table 6.5, retransmitted data poses a challenge to both systems, and the clear dominance of the model employing Conv-TasNet small from the previous experiment has diminished. On the other hand, despite the significantly smaller size of the proposed model, it still performs on par with or slightly better than the mask-predicting model. Comparing *dev retr SRE* and *dev retr CE* (its clean-enrollment counterpart) as well as *eval retr v1 SRE* with *eval retr CE*, the results suggest that enrollment recordings that are clean lead to better results on average. It holds for both systems. It is expected behavior, but we take the liberty to emphasize it as it is appealing for practical scenarios. As expected, compact microphone arrays that do not suffer from outlier signals of bad quality provide considerably better results. This outcome stems from the comparison of *eval retr v1 SRE* and *eval retr v2 SRE*. We note that results on the other MultiSV protocols can be found in [Mošner et al., 2022b].

### Joint Model Fine-Tuning

Since loss functions optimized by speech enhancement models might not be directly related to the quality of speaker embeddings, we adopted a previously explored front-end fine-tuning scheme. We join both separately trained models and optimize the speaker-discriminative AM softmax loss. This is doable as all components (including beamforming) of the models are differentiable. During the fine-tuning phase, the ResNet extractor is fixed, and only the front-end parameters are updated. We re-utilize data used for separate front-end training, MultiSV, as it also includes speaker labels. Before this fine-tuning, the last layer was redefined and trained since the number of speakers differs from the number of speakers in the embedding extractor training set.

The lower part of Table 6.5 shows considerable improvements by fine-tuning. Our system with Conv-TasNet small benefits from this phase more, as the average relative improvement is 8.0% compared to 6.4% for the mask-predictor-based model. Overall, our model yields the best results across the board. This outcome suggests its power and generalizability.

#### Discussion

Apart from the presented approach to SCM estimation, we also experimented with a more straightforward method directly utilizing Conv-TasNet small outputs to estimate speech and noise SCMs similarly to [Ochiai et al., 2020], i.e., without masking. We will elaborate on it in the next section. Even though we obtained 1.23% EER, 0.155 minDCF on *dev simu*, and 1.99% EER, 0.179 minDCF on *eval simu*, the proposed approach consistently outperforms it, especially on retransmitted data.

In this study, Conv-TasNet small enhances individual channels independently. Utilization of multiple channels to enhance one of them might be helpful. Although a simple concatenation of channels did not improve results in our preliminary experiments, a more complex approach (presented in Section 5.7.1) that turned out feasible in speech separation will be explored in the following section.

It is noteworthy that the data used for fine-tuning is simulated. During this phase, speaker labels suffice, and no speech-noise decomposition is required. We hypothesize that multi-channel real training data with speaker labels could boost performance even more.

### 6.4.5 Framework Analysis: Front-Ends and Joint Fine-Tuning

In Sections 6.4.3 and 6.4.4, we focused on establishing the framework of cascaded multi-channel front-end and single-channel embedding extraction by exploring various design choices. Due to different backends and datasets, it is not straightforward to draw comparative conclusions. Throughout our explorations, we converged to the MultiSV and MultiSV2 datasets used in this section to ease comparison.

Building upon our empirical observations, we will employ the MVDR beamformer in all the following experiments. In order to limit the number of potential framework instances, we will employ an updated ResNet34 embedding extractor, noting that more powerful ones could be built by leveraging novel architectures and massive training corpora. However, our embedding extractor is required to be sufficiently compact to leave enough memory for front-end models, especially for time-domain models that employ small window sizes (e.g., four or fewer samples), which are memory-demanding due to huge tensors propagated through networks. At the same time, the embedding extractor is reasonably performant, as we will show later. In the light of explorations dedicated solely to separation in Section 5.6, we will dedicate most of the space to a comprehensive analysis of the impact of front-end variants on SV performance. Our ultimate goal is to fine-tune the joint models together using the final speaker-discriminative objective. Although we scratched the surface of this topic in Sections 6.4.3 and 6.4.4, we will provide a thorough analysis with various front-ends and fine-tuning approaches.

#### Embedding Extractor

We employ a ResNet34-based model as the extractor of speaker embeddings following the architecture in Table 2.2, i.e., similar to that in [Zeinali et al., 2019], with modifications provided in this paragraph. Compared to the ResNet34 extractor in Section 6.4.4 (but

Table 6.6: Speaker verification performance evaluated on various single-channel datasets. For MultiSV, the performance corresponds to the average of four single-channel results.

| Evaluation set | EER | minDCF |
|---|---|---|
| VoxCeleb1-O | 1.08 | 0.086 |
| VoxCeleb1-E | 0.93 | 0.106 |
| VoxCeleb1-H | 2.13 | 0.210 |
| MultiSV CE | 4.55 | 0.276 |
| MultiSV SRE | 4.70 | 0.281 |
| MultiSV MRE | 4.65 | 0.357 |
| MultiSV MRE_hard | 6.31 | 0.473 |

in accordance with Table 2.2), we use *fbank* feature vectors with 80 dimensions, where window size and shift are 25 ms and 10 ms, respectively. We follow the full pre-activation structure of the residual building blocks [He et al., 2016b]. In the first and second stages, we augment them by incorporating the Squeeze-and-Excitation (SE) [Hu et al., 2020a] units right after the last convolutional layer (and before the residual connection). The ratio of a dimensionality reduction layer within the SE is set to 4. The frame-level part of the model is followed by a standard statistics pooling and projection to a 256-dimensional embedding space. The embedding extractor comprises 14.64 M parameters, and it is trained to optimize additive angular margin (AAM) loss [Deng et al., 2019] with a scale of 30 and a margin scheduled to rise from 0.1 to 0.3.

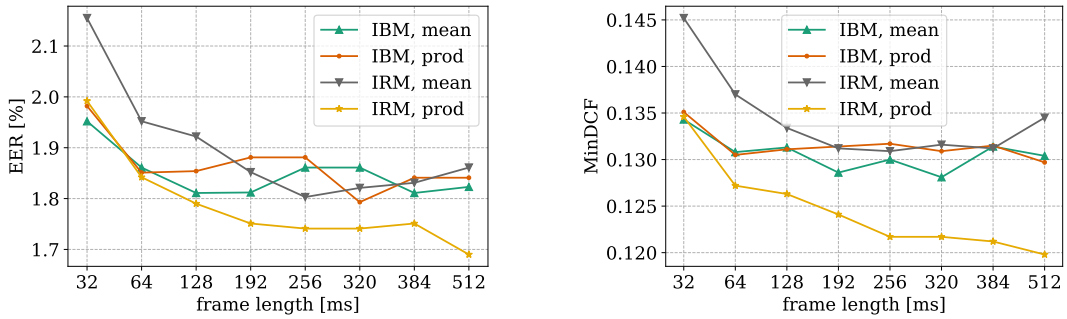
#### Single-Channel Speaker Verification

In order to demonstrate the performance of our embedding extractor, we first evaluated it on standard verification trial sets, VoxCeleb1 O, E, and H (see Section 4.1 and [Nagrani et al., 2017]). Table 6.6 presents the results. We note that recent systems can yield better results, but achieving them was not our primary goal. We showcase concepts and trends while keeping models' sizes within reasonable limits. Nevertheless, given our system's architecture and the fact that neither score normalization nor large-margin fine-tuning were employed [Thienpondt et al., 2021], it still represents a competitive model.

Table 6.6 also reports the performance of the same embedding extractor on single-channel versions of the MultiSV *retr eval v1* trial lists. Each number represents the average of four results obtained using trials created by systematically selecting individual channels from the original microphone arrays. For CE and SRE, where each trial consists of a single-channel enrollment and a 4-channel test recording (see Section 4.3.3), the single-channel variants were created by pairing the original enrollment with each channel of the test side, one at a time. For MRE and MRE_hard, which contain 4-channel enrollment and test recordings, the single-channel versions were formed by pairing corresponding channels from the enrollment and test recordings across the arrays. These results provide a reference point for evaluating the effectiveness of the multi-channel pre-processing presented later and highlight the difficulty of the dataset.

#### Analysis Window Size and Shift

The beamformers in approaches described in Sections 6.4.3 and 6.4.4 processed audio transformed by STFT with an analysis window size and shift of 64 and 16 ms, respectively.



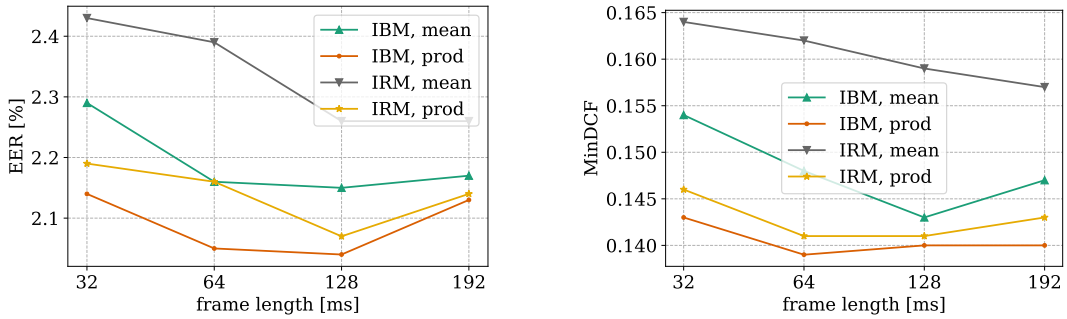

(a) EER (b) minDCF

Figure 6.6: SV metrics obtained with outputs of the oracle MVDR beamformer as a function of the frame length used in STFT. Different curves correspond to different combinations of masking and pooling approaches (prod stands for product pooling). The frame step is always set to a quarter of the frame length, which is widely accepted in the literature. MultiSV simu eval data was used for evaluation.

However, Sections 5.4 and 5.6.1 revealed a significant impact of the two hyperparameters on the separation performance. With this in mind, we also aim to assess the dependence of the SV on the window properties. Before using trainable front-ends, we will consider an oracle beamformer as in Section 5.4. Along with the window size and shift, we also intend to quantify the effect of mask type (IBM, IRM) and pooling of masks (mean, product).

In order to conduct such controlled experiments, we use verification trials of the simulated evaluation MultiSV set (*eval simu*). The simulated recordings allow us to compute oracle beamforming weights by exactly expressing IBMs or IRMs. We recall that only the test parts of trials are multi-channel. Enrollment recordings contain single-channel clean or reverberated speech.

The results in terms of EER and minDCF are provided in Figures 6.6a and 6.6b, respectively. Generally, the curve trends are downward, meaning speaker verification benefits from increased window sizes. We recall that in terms of source separation, we drew the same conclusions for separation metrics (see Figure 5.2). Therefore, we can conclude that significant improvements in separation ability translate to improved speaker verification measured in the oracle conditions. In contrast to Figure 5.2, we also note that speaker verification results are much noisier. This corresponds to the fact that no trainable model participated in the evaluation in the case of separation. On the other hand, a pre-trained speaker embedding extractor provides embeddings for enrollment or beamformed test recordings in SV. Focusing on IBM, we can draw similar outcomes as in Section 5.4: the pooling type is not a critical hyperparameter for such a mask type. We initially observe a gain when increasing the window length (and shift), but the curve quickly stabilizes, and the results continue fluctuating. The pooling type becomes vital when substituting IBMs with IRMs. Although the trends are similar, combining IRMs with product pooling is more performant. At the same time, such a combination yields the best overall results, reinforcing our conclusions in source separation in Section 5.4.




(a) EER (b) minDCF

Figure 6.7: SV metrics obtained with outputs of the mask-based MVDR beamformer (employing neural-network mask predictors) as a function of the frame length used in STFT and by mask estimators. Different curves correspond to different combinations of training targets (IBM or IRM masks) and pooling approaches (prod stands for product pooling). The frame step is always set to a quarter of the frame length. MultiSV simu eval data was used for evaluation.

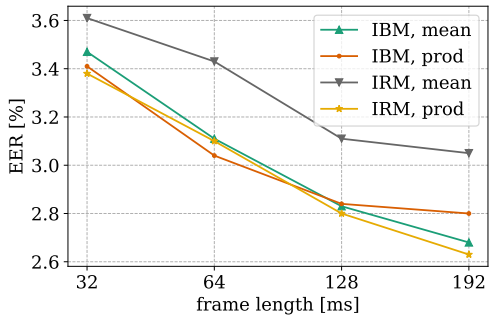
### Direct Mask Prediction

In this section, we will focus on analyzing direct mask prediction (Figure 5.4a), which we employed in Section 6.4.3 for the GEV beamformer. This exploration is motivated by Section 5.6.1 (dedicated to source separation) and the previous subsection, where we analyzed ideal masks and pooling type.

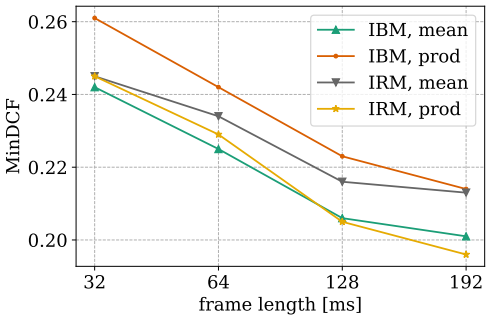
We intend to study how approximations of IBMs and IRMs, and their aggregation, affect speaker verification performance. To this end, we employ the same backbone LSTM-based mask estimators as in Section 5.6.1 trained with either IBM or IRM targets, optimizing binary cross-entropy or mean-squared error, respectively. We also follow the estimation of SCMs defined in (5.10).

The frame-dependence results are presented in Figures 6.7a (EER) and 6.7b (minDCF). First, we notice that mean pooling paired with a ratio-mask predictor performs the worst. This outcome correlates with what we drew for the oracle beamformer in Figure 6.6. The second noticeable trend relates to the pooling type. Regardless of the predictor, product pooling helps to enhance performance. Focusing on product pooling, the IBM predictor performs better (which was not the case in the oracle experiments). However, we stress that in addition to the effect of pooling, the optimization of the mask estimators plays an important role. Moreover, the difference between the two corresponding curves is barely significant.

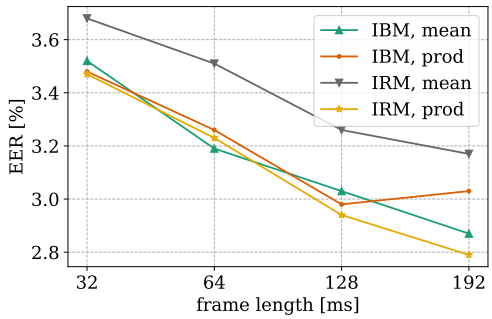
The presented results were obtained using the simulated MultiSV data (eval simu). In order to obtain a complex view of the behavior, we also experimented with all the protocols of the retransmitted MultiSV. We provide the outcomes in Figure 6.8. We confirm that the IRM prediction and mean pooling combination is the worst in terms of EER. Recalling separation results in Figure 5.5, we stress that poor verification performance indeed correlates with poor separation performance. Although the IBM estimator with product pooling may provide the best results under some circumstances, it does not behave consistently (which we also observed for mean SDRi and STOI metrics when dealing with



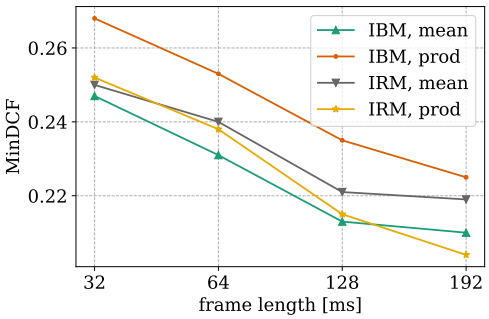

(a) CE, EER



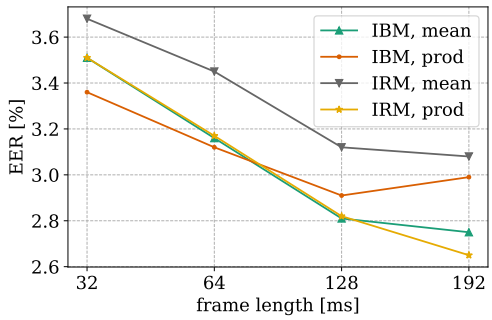

(b) CE, minDCF



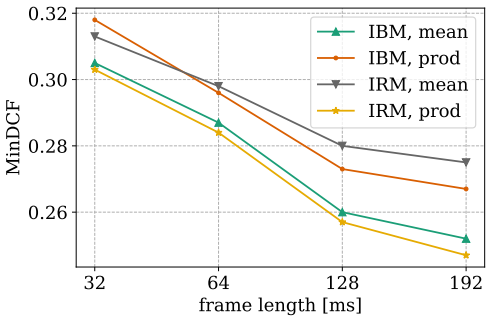

(c) SRE, EER



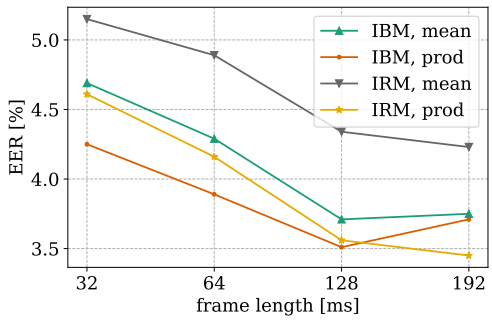

(d) SRE, minDCF



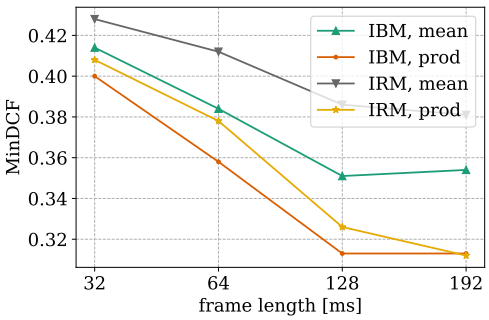

(e) MRE, EER




(f) MRE, minDCF




(g) MRE_hard, EER




(h) MRE_hard, minDCF

Figure 6.8: SV metrics obtained with outputs of the mask-based MVDR beamformer (employing neural-network mask predictors) as a function of the frame length in STFT. Different curves correspond to different combinations of training targets (IBM or IRM masks) and pooling approaches. MultiSV retr eval data was used for evaluation.

Table 6.7: Comparison of SV results when employing indirect mask prediction for mask-based beamforming with mask-less beamforming and the network outputs (✗ in the beamforming column). In all the approaches, time-domain Conv-TasNet and DPTNet were used as backbone models. We used a frame size of 128 ms and a shift of 32 ms in beamforming. The evaluation was performed on MultiSV eval retr v1.

| Base model | Beamforming | CE | | SRE | | MRE | | MRE_hard | |
|---|---|---|---|---|---|---|---|---|---|
| | | EER | minDCF | EER | minDCF | EER | minDCF | EER | minDCF |
| Conv-TasNet | mask-based | 2.55 | 0.202 | 2.70 | 0.213 | 2.88 | 0.274 | 3.62 | 0.332 |
| | mask-less | 3.16 | 0.224 | 3.34 | 0.237 | 3.15 | 0.269 | 4.03 | 0.330 |
| | ✗ | 5.62 | 0.311 | 5.88 | 0.318 | 6.06 | 0.423 | 9.71 | 0.626 |
| DPTNet | mask-based | 2.38 | 0.198 | 2.56 | 0.211 | 2.62 | 0.256 | 3.16 | 0.295 |
| | mask-less | 3.03 | 0.224 | 3.09 | 0.235 | 3.13 | 0.279 | 4.22 | 0.333 |
| | ✗ | 5.50 | 0.307 | 5.64 | 0.313 | 5.67 | 0.408 | 8.22 | 0.590 |

the separation performance). On the other hand, the behavior of IRM predictor with product pooling is stable across various settings and metrics.

Finally, we can conclude (and strengthen the conclusions of Section 5.6.1) that the direct mask prediction benefits the most from the IRM predictors where the masks are pooled by product. In our scenarios with static sources, speaker verification (and source separation) performance improves with increasing frame size and shift. For reasons matching those provided in Section 5.6.1, we will further use a frame size of 128 ms and a shift of 32 ms.

### Indirect Mask Prediction and Mask-Less Beamforming

This section aims to analyze time-domain separation models (successors of the time-frequency ones) in the context of SCM prediction for beamforming and subsequent speaker verification. As in Sections 5.6.2 and 5.6.3, we will explore indirect mask-prediction (Figure 5.4b) and mask-less beamforming using both Conv-TasNet and DPTNet models.

First, we will tackle indirect mask prediction. In Section 5.6.2 (about speech separation), we observed that DPTNet could provide better speech and noise predictions for simulated recordings than Conv-TasNet as measured by separation metrics. However, when using its outputs for indirect mask prediction and subsequent beamforming, we observed no benefit from using DPTNet. Interestingly, when we employ the same underlying separation networks that support SCM estimation to process more difficult retransmitted data, DPTNet manifests superiority over Conv-TasNet. We drew this conclusion by comparing SV results in the *mask-based* rows of Table 6.7. Importantly, as the data becomes harder (toward the MRE_hard protocol), the gap between Conv-TasNet and DPTNet becomes more prominent. These observations suggest that some conclusions based on simulated data might not be reliable and should be treated cautiously.

Table 6.7 also shows results obtained when using separated time-domain signals to estimate SCMs without masking according to (5.14) (mask-less beamforming). The speech separation experiments in Section 5.6.3 showed that such an approach to beamforming provides better mean SDRi but worse STOI than mask-based beamforming. One could expect that better mean SDRi will translate to better SV performance (as speech intelligibility is arguably less important for SV). However, the SV results with mask-less beamforming are consistently worse than mask-based beamforming. We hypothesize that this is because of signal distortions (introduced by neural networks) that are directly propagated to SCM estimation. The beamformer then combines multi-channel mixture signals utilizing SCMs

Table 6.8: Comparison of pre-processing alternatives with fixed embedding extractor. The verification results were obtained using MultiSV evaluation protocols with retransmitted data. Separation metrics were computed using separation network outputs (or fused network and beamformer outputs when indicated) on a simulated MultiSV evaluation set. The *RCA* column indicates the utilization of the reference channel attention module. When the *RCA adapt.* is on, the whole multi-channel separation model, including the RCA, was adapted on multi-channel data. *Fus.* means fusion of beamformer and network outputs.

| Base model | | RCA | RCA adapt. | fus. | CE | | SRE | | MRE | | MRE_hard | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | EER | minDCF | EER | minDCF | EER | minDCF | EER | minDCF |
| Diff-Filter [Dowerah et al., 2023a] | A | ✗ | ✗ | ✗ | — | — | — | — | 3.07 | — | 3.19 | — |
| FaSNet TAC | B | ✗ | ✗ | ✗ | 5.02 | 0.300 | 5.14 | 0.307 | 5.13 | 0.400 | 7.68 | 0.565 |
| Mask estim. | C | ✗ | ✗ | ✗ | 2.80 | 0.205 | 2.94 | 0.215 | 2.82 | 0.257 | 3.56 | 0.326 |
| Conv-TasNet | D | ✗ | ✗ | ✗ | 2.55 | 0.202 | 2.70 | 0.213 | 2.88 | 0.274 | 3.62 | 0.332 |
| | E | ✗ | ✗ | ✓ | 3.62 | 0.228 | 3.85 | 0.232 | 4.06 | 0.326 | 5.36 | 0.433 |
| | F | ✓ | ✗ | ✗ | 2.54 | **0.198** | 2.69 | 0.210 | 2.64 | 0.252 | 3.27 | 0.287 |
| | G | ✓ | ✗ | ✓ | 3.82 | 0.235 | 4.03 | 0.242 | 4.11 | 0.323 | 5.17 | 0.427 |
| | H | ✓ | ✓ | ✗ | **2.43** | 0.199 | **2.67** | **0.209** | **2.63** | **0.251** | **3.26** | **0.281** |
| | I | ✓ | ✓ | ✓ | 3.80 | 0.235 | 4.02 | 0.242 | 3.94 | 0.321 | 4.81 | 0.425 |
| DPTNet | J | ✗ | ✗ | ✗ | **2.38** | 0.198 | **2.56** | 0.211 | 2.62 | 0.256 | 3.16 | 0.295 |
| | K | ✗ | ✗ | ✓ | 3.51 | 0.221 | 3.68 | 0.231 | 3.64 | 0.304 | 4.55 | 0.391 |
| | L | ✓ | ✗ | ✗ | 2.43 | 0.198 | **2.56** | 0.211 | 2.54 | 0.256 | 3.06 | 0.293 |
| | M | ✓ | ✗ | ✓ | 3.49 | 0.224 | 3.66 | 0.231 | 3.59 | 0.307 | 4.52 | 0.386 |
| | N | ✓ | ✓ | ✗ | 2.43 | **0.196** | **2.56** | **0.209** | **2.50** | **0.250** | **3.01** | **0.283** |
| | O | ✓ | ✓ | ✓ | 3.47 | 0.226 | 3.70 | 0.230 | 3.64 | 0.304 | 4.50 | 0.388 |

that carry statistics of neural-network-processed data. Due to the better results attained with mask-based beamforming, we will employ it in the experiments in the next sections.

Finally, we aim to strengthen the justification of our cascaded design of beamforming and embedding extraction. We experimented with extracting embeddings using reference-microphone speech separated directly by network-based separators. Per the results in Table 6.7, the performance is considerably worse than mask-less beamforming across the board when employing direct predictions.

### Extending Single-Channel Separators with the RCA Combiner

In this section, we focus on an in-depth analysis of the effects of the pre-processing, focusing on our original RCA combiner (Section 5.7.1). We note that the pre-processing networks are still trained in isolation, i.e., not to maximize speaker discrimination explicitly, as we keep the embedding extractor intact.

Before focusing on the RCA, we will make a note on multi-channel processing in general. Comparing Table 6.6 (single-channel results) and Table 6.8, presenting results on the multi-channel protocols of MultiSV eval retr, we observe that all variants of our framework provide better performance in all measured verification metrics. This reinforces the benefit of using multiple microphones in far-field speaker verification. We also note that masks predicted indirectly by DPTNet (J) lead to better SV performance than those directly predicted by the LSTM-based model (C). However, this should not be interpreted as a definitive advantage of indirect prediction since the observed differences also stem from architectural differences (notice that the Conv-TasNet-based system, D, outperforms C only in half of the cases).

Until now, only the beamforming module was responsible for integrating information across multiple channels. Aiming to utilize the availability of multiple channels earlier in the pipeline, we further investigate a multi-channel separation that takes a reference channel (for which it predicts sources) and other channels as conditioning (Figure 5.4b). We achieve this by extending per-channel operating models D and J by incorporating the RCA combiner. We freeze the pre-trained single-channel separators and update only the RCA on multi-channel MultiSV training data. The converged models are used to predict the sources for all the channels (conditioned on the others), and the computation of the beamforming weights is analogous to the previous experiments with mask-based beamforming. The extended models (F, L) yield SV results comparable to those of their original counterparts (D, J) on the CE and SRE protocols. We recall that multi-channel processing is applied to the test sides of trials only, while enrollment recordings are single-channel. On the other hand, the benefits of the RCA in multi-channel SV start taking effect when processing more challenging retransmitted and, importantly, multi-channel (enrollment and test) data (MRE and MRE_hard). With the Conv-TasNet-based model, we observed a relative EER improvement of 9.7% on MRE_hard by incorporating RCA. Strong DPTNet benefited less and provided a relative improvement of 3.2% in the same settings.

Eventually, having multi-channel separators (F, L) obtained by introducing and training RCA, we attempted further adaptation on multi-channel data by joint (separator and RCA) training, resulting in models H and N. We observed similar performance after adaptation, suggesting that mere RCA training is sufficient and fast.

We also reiterate the issue of direct separators degrading downstream task performance highlighted in the previous section, reinforcing the architecture design of our framework. To this end, we employed a direct multi-channel separator. Then, the system B is a sequence of neural-based multi-channel FaSNet TAC [Luo et al., 2019] and our embedding extractor. FaSNet TAC was trained on the same data as all other separators (using all 4 channels of the MultiSV training part). Even though the pre-processing model performs a segment-wise filter-and-sum operation, the filters are estimated by a neural network, hence introducing distortions. Although the separator provides competitive separation results (per Table 5.6), a degradation in verification performance is noticeable.

### Network and Beamformer Outputs Fusion

Following Section 5.7.3, we explore the fusion of beamformer and separation network outputs. Compared to the separation experiments, we are interested solely in the speech component $\hat{\mathbf{x}}^{(\text{fus})}$ in (5.22). Table 6.8 shows the results obtained for all the time-domain variants of separation networks (✓ in column *fus.*). We noticed significant deterioration resulting from the fusion. We hypothesize that it is related to the aforementioned phenomenon: By fusing beamformer output and neural network output, we introduce non-linear distortions to the signal, which is passed to the embedding extractor, whose performance consequently drops.

Observing trends in speaker verification performance (and also unsatisfactory separation results when using a frame size of 128 ms for beamformer) by fusing beamformer and network outputs, we performed the following analysis using Conv-TasNet as an underlying separation model. In our study [Mošner et al., 2023], we observed gains in separation ability for 6-microphone arrays. In MultiSV, the arrays comprise only four microphones. Therefore, we created new evaluation data using a procedure similar to generating the MultiSV simulated evaluation set. We varied the number of microphones to test dependence on the

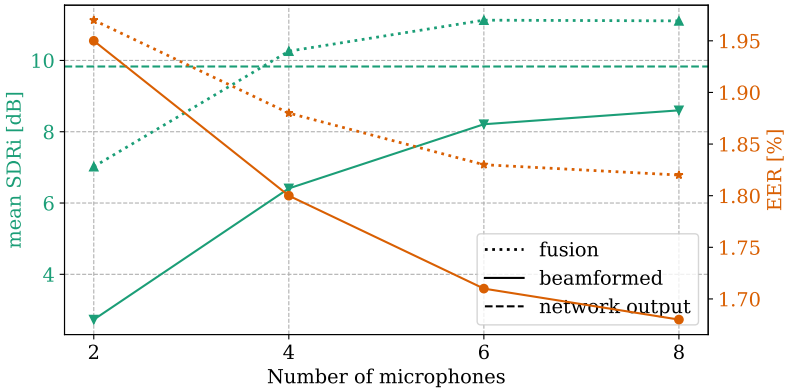



Figure 6.9: Separation capability (mean SDRi) and speaker verification performance (EER) as functions of sensor count in microphone arrays. The simulation was based on speech source data of the simulated MultiSV evaluation set. Conv-TasNet was used as a separation model. Frame size of STFT for broadband MVDR beamformer was 128 ms.

channel count. Having arrays of 2, 4, 6, and 8 microphones, we measured EER (using single channel enrollment) as well as mean SDRi for two outputs — beamformer only and fusion. The results are presented in Figure 6.9. As expected, beamforming provides better-separated outputs with more microphones. However, even with eight sensors, it still does not reach the mean SDRi of the single-channel Conv-TasNet. Fusion has the positive effect of consistently improving mean SDRi over the beamformer output. We observe that we reach a borderline, where the fusion outputs approach the network outputs in terms of mean SDRi, when using four microphones. With six or more microphones, gains in separation performance by fusion are observable, supporting outcomes in [Mošner et al., 2023].

The improved beamformer outputs translate to a better EER. The best one is achieved with eight sensors. When we feed the embedding extractor with fused signals, the performance also improves with more microphones. However, it is still noticeably worse than that achieved with beamformer-only inputs, supporting our previous findings.

### Joint Model Fine-Tuning on Large Data

Finally, allowed by the differentiability of components of all framework instances, we explore joint training using our extended MultiSV2 training data (Section 4.4). Noting that we experimented with other alternatives, we present a strategy that we empirically found viable: In the first step, we fine-tune only the embedding extractor using the AAM loss, keeping the front-end fixed. After the first step, we make both the front-end and embedding extractor trainable. Starting from the checkpoint obtained in the first step, we continue training and keep the same objective.

We recall that fine-tuning with a speaker discriminative loss was employed already in Sections 6.4.3 and 6.4.4. Only the front-end models were updated in those cases to provide outputs suitable for the given embedding extractors. Our ultimate goal is to fine-tune all parameters of the joint model. It requires a reasonably-sized multi-channel dataset with speaker labels. In Section 6.4.3, we used retransmitted data from the VOiCES dataset; in Section 6.4.4, the fine-tuning was facilitated by utilizing simulated MultiSV. None of these corpora is large enough to enable updating the embedding extractor. Therefore, we resorted to creating the MultiSV2 dataset (see Section 4.4) by relaxing the SNR constraint

Table 6.9: Fine-tuning components of joined models with a common speaker-discriminative (AAM) objective. *Emb.* in the fine-tuning column means that only the embedding extractor was fine-tuned using the MultiSV2 data. *Sep.+emb.* signifies joint fine-tuning of both the separator and speaker embedding extractor.

| Separator model | fine-tuning | | MRE | | MRE_hard | |
|---|---|---|---|---|---|---|
| | | | EER | minDCF | EER | minDCF |
| Conv-TasNet | ✗ | D | 2.88 | 0.274 | 3.62 | 0.332 |
| | emb. | D.2 | 2.45 | 0.199 | 2.77 | 0.248 |
| | sep.+emb. | D.3 | **2.23** | **0.194** | **2.55** | **0.233** |
| Conv-TasNet + RCA | ✗ | H | 2.63 | 0.251 | 3.26 | 0.281 |
| | emb. | H.2 | 2.44 | 0.212 | 2.70 | 0.250 |
| | sep.+emb. | H.3 | **2.18** | **0.188** | **2.45** | **0.224** |
| DPTNet | ✗ | J | 2.62 | 0.256 | 3.16 | 0.295 |
| | emb. | J.2 | 2.40 | 0.209 | 2.68 | 0.246 |
| | sep.+emb. | J.3 | **2.25** | **0.195** | **2.48** | **0.230** |
| DPTNet + RCA | ✗ | N | 2.50 | 0.250 | 3.01 | 0.283 |
| | emb. | N.2 | 2.35 | 0.208 | 2.58 | 0.243 |
| | sep.+emb. | N.3 | **2.21** | **0.189** | **2.44** | **0.226** |

on the speech source recordings compared to MultiSV. As a result, the speech component of the simulated data may contain traces of noise (different from the artificially added one) compared to VOiCES and simulated MultiSV. We accepted this trade-off to obtain a sufficient volume of data (1,157 hours). We note that we also attempted to fine-tune only the front-end with this dataset (as in Sections 6.4.3 and 6.4.4). However, we did not achieve noticeable improvements despite an increase in the number of training examples. We hypothesize that it is because of the noise leakage into the source speech, since such noise does not fit the arrangement of a simulated environment. Then, it is not possible to suppress distractors whose spatial location coincides with the location of the source of interest. Therefore, when updating only the front-end, arguably, two competing objectives prevent gaining improvement: On one hand, better separation of speech is achieved by assigning noise traces and additive noise to one output. On the other hand, the noise output is used to estimate the noise SCM. Therefore, assigning noise traces to the non-speech channel results in suppression of speech due to colocation.

Intending to provide a comprehensive analysis, we take models based on both Conv-TasNet and DPTNet separators (systems D, J in Table 6.8) and their multi-channel alternatives with RCA (systems H, N) as starting points. The results of the two fine-tuning steps are presented in Table 6.9. We can conclude that the same trends hold for all the starting models: Embedding extractor fine-tuning significantly boosts performance. The following step (*sep.+emb.*), also adjusting the pre-processing, provides another less pronounced improvement.

Focusing on the embedding extractor fine-tuning and a specific separator backbone (Conv-TasNet or DPTNet), we observe that the models with or without RCA perform comparably (i.e., D.2 is comparable with H.2; J.2 is comparable with N.2). It suggests that the embedding extractor adaptation on beamformed inputs of similar quality yields comparably performant models.

Comparing Conv-TasNet- and DPTNet-based models, the former benefits more from the extractor fine-tuning as the relative improvements in results are higher. Nevertheless, the DPTNet-based systems (J.2 and N.2) still perform better due to a better starting point (J and N).

Concentrating on the second stage, we conclude that both Conv-TasNet- and DPTNet-based models achieve similar performance (i.e., D.3 is comparable with J.3; H.3 is comparable with N.3). Moreover, models incorporating the RCA module for processing multi-channel inputs (H.3 and N.3) tend to benefit more from the second stage compared to their non-RCA counterparts, representing overall the best performing models.

Last but not least, we observe that all the fine-tuned instances of our framework significantly outperform Diff-Filter (see system A in Table 6.8). Diff-Filter also represents a model comprising multi-channel pre-processing and a single-channel embedding extractor. Finally, it is jointly updated using a speaker-verification-related objective.

## 6.5 Summary

We dedicated this chapter to cascaded systems in speaker verification. Initially, we dealt with far-field scenarios, where reverberation is inevitable, without additive noise. We showed that by employing WPE dereverberation, mask-based GEV beamforming, and careful selection of training data for PLDA, the accuracy can approach that achieved on corresponding clean data. The system did not even use neural network speaker embedding extractors, but i-vectors to summarize recordings.

Having gained insights about how to combat reverberation in multi-channel speaker verification successfully, we moved to more difficult and realistic scenarios, where the source of distraction (noise) is present. To tackle them, we established a framework that we developed in Sections 6.4.3 and 6.4.4, and benchmarked its instances in Section 6.4.5. Regarding multi-channel pre-processing, we explored direct and indirect mask prediction for mask-based and direct mask-less beamforming. To this end, we employed various speech separation models reflecting progress in the source separation field. Then, we extended the time-domain separators with our RCA, making predictions for a specific microphone conditioned on signals from other microphones. In the beamforming stage, we explored broadband versions of two well-known beamformers, GEV and MVDR. In the single-channel part of the framework, we tested multiple neural networks: an extended version of x-vectors (E-TDNN) and alternations of ResNet34, where we used various training objective functions. Finally, in order to obtain verification scores, we adopted different backends, namely PLDA and cosine similarity scoring.

Both front-end and embedding extractors pose conditions on the training data. In order to satisfy them, we explored and built different variants of data for speech separation models. Our investigations converged to creating our dataset, MultiSV, which we eventually used for benchmarking and comparing framework instances.

Since front-end and embedding extractors optimize different criteria in the first place, we aimed to make all the components of the whole pipeline differentiable so that the whole final model (or its part) can be updated by minimizing the ultimate speaker-identity-related loss function. To this end, we were concerned with reformulating GEV using Cholesky decomposition. Fine-tuning with the final objective also requires data on specific qualities, such as the multiplicity of channels and speaker labels. As in the case of the front-end training, we explored different ways until landing on MultiSV2.

In our analyses, we made multiple key observations. We emphasized the utility of multi-channel systems compared to single-channel ones. We confirmed that a significant improvement in the front end positively affects the whole system's performance. Motivated by the observed impact of the size of the analysis window for broadband beamforming in speech separation, we conducted experiments suggesting that SV also depends on the frame size. Although maskless beamforming can deliver better separation measured by mean SDRi, we showed it lags behind mask-based beamforming when employing the outputs for embedding extraction. Our analysis also included fusing the beamformer and separation network outputs. We concluded that, in contrast to separation improvements, it led to SV results deterioration. Incorporating our RCA combiner into time-domain separators indeed boosted SV performance, confirming the utility of using available channels directly in speech separation. Finally, we presented a viable strategy to eventually update all the parameters of the joint models with a speaker-discriminative objective, leading to gains in performance. Our best fine-tuned models employed the RCA combiner.

Despite the advantages of the presented framework, it is rather complex, which makes experimentation and completion of the steps leading to the best results demanding. Moreover, the performance is bounded by a pre-defined fixed beamforming formulation. In light of these issues (but still inspired by achieved results), we will explore multi-channel front-ends using the strength of neural networks to perform a combination of channels without explicit separation supervision, leading to end-to-end multi-channel embedding extractors.

# Chapter 7

# Multi-Channel Speaker Verification Building on Foundation Speech Models

In Chapter 6, we dealt with cascaded multi-channel SV systems comprising pre-processing and speaker verification components. Despite better interpretability and modularity of such models, they are limited by the loose correlation between the enhancement and downstream objectives [Shon et al., 2019]. Joint fine-tuning can improve performance (see [Mošner et al., 2022a, Dowerah et al., 2023a] and Section 6.4.5), but only partly alleviates this limitation. Another limitation is that spatial information can be leveraged only by the pre-processing stage, and enhancement errors get propagated to the downstream model.

Observing the downsides of the cascaded systems, we aim to explore a different paradigm. Motivated by the advent of large speech representation models following a self-supervised learning (SSL) scheme (foundation models), we will explore ways to use them in multi-channel speaker verification, effectively skipping the pre-processing stage. The challenge is that they are usually pre-trained using single-channel data. Therefore, there are two fundamental questions that we will tackle: 1) How to extend SSL models to multi-channel data? 2) How to leverage extended models for speaker verification?

## 7.1 Foundation Models and Multi-Channel Data

Self-supervised learning has attracted significant attention in speech processing [Baevski et al., 2020, Hsu et al., 2021, Chen et al., 2022a]. It allows for training models on large-scale audio data without labels. SSL models are typically trained on clean data, which makes them less suitable for signals corrupted by room conditions. Recently, studies extending them to different flavours of data have been published [Chen et al., 2022a, Fazel-Zarandi and Hsu, 2023]. For instance, WavLM [Chen et al., 2022a], among other modifications over HuBERT [Hsu et al., 2021], was designed to be noise-robust. The resulting general representations of speech were shown to be useful for many tasks (including SV) in the SUPERB benchmark [Yang et al., 2021], where the SSL model is frozen. Recent studies [Peng et al., 2023a, Chen et al., 2022a, Chen et al., 2022b, Novoselov et al., 2023] have demonstrated an impressive SV performance when it is fine-tuned with the downstream models.

Commonly available foundation models are single-channel by design. Inspired by the robust performance of models leveraging pre-trained speech representations, attempts to

utilize them in multi-channel settings have emerged. All the top CHiME-7 Challenge systems [Cornell et al., 2023], as well as [Masuyama et al., 2023a, Masuyama et al., 2023b], are based on cascading multi-channel beamforming/enhancement, the SSL model, and (end-to-end) single-channel automatic speech recognition.

A few works have explicitly extended the SSL framework to multi-channel models. Spatial HuBERT [Dimitriadis et al., 2023] is an SSL model built upon WavLM. It extends the CNN encoder to its multi-channel version. Multi-channel AV-wav2vec2 [Zhu et al., 2024] is an SSL model for multi-channel, multi-modal data, which fuses channels and modalities before the Transformer encoder blocks. None of the mentioned models was publicly released.

## 7.2 Multi-Channel Extension of Pre-Trained Models

We call our approach METRO (Multi-channel ExTension of pre-tRained mOdels). It is a general technique to extend single-channel SSL models to multi-channel data. Despite focusing on SV in this paper, our design is applicable to other speech tasks. It replicates the original blocks to perform parallel processing of channels up to a certain level. Then, the channels are fused, and the original SSL architecture continues. Importantly, we introduce new modules after each parallel block to promote information exchange across channels. We stress the importance of properly initializing these new modules to avoid corrupting the forwarded data. Our approach achieves considerably better results than previously presented models on the MultiSV corpus. Our holistic solution for multi-channel SV builds upon the following concepts:

1. Inspired by its robust speech representations, we employ WavLM [Chen et al., 2022a] as a backbone for our model.

2. Among various approaches to using SSL features for SV, we opted for multi-head factorized attentive pooling (MHFA) [Peng et al., 2023a] (described in Section 2.4.1) as it is lightweight yet competitive.

3. We adopt the idea of extending a single-channel model while fusing channel-wise information at the frame level [Liang et al., 2022].

4. Multi-channel architectures interleaving cross-channel and temporal per-channel processing have become successful in speech separation [Wang et al., 2020, Wang et al., 2021a] and diarization [Horiguchi et al., 2022b]. We reach the same structure by introducing new modules to SSL models (see Figure 7.1).

The approach we develop in this chapter falls within the third category of designs described in Section 6.2, which extends single-channel models. We recall that such models usually start from a strong single-channel speaker embedding extractor, processing channels independently up to a specific layer. Subsequently, modules that fuse information across channels are introduced. The information can be fused at the utterance level [Cai and Li, 2021, Liang et al., 2021] or at the frame level [Liang et al., 2022], where the latter was shown to be superior.

### 7.2.1 Method

To extend SSL models to multi-channel data, we replicate the original blocks (CNN encoder, Transformer encoder blocks) to allow for parallel processing. The same transformation is

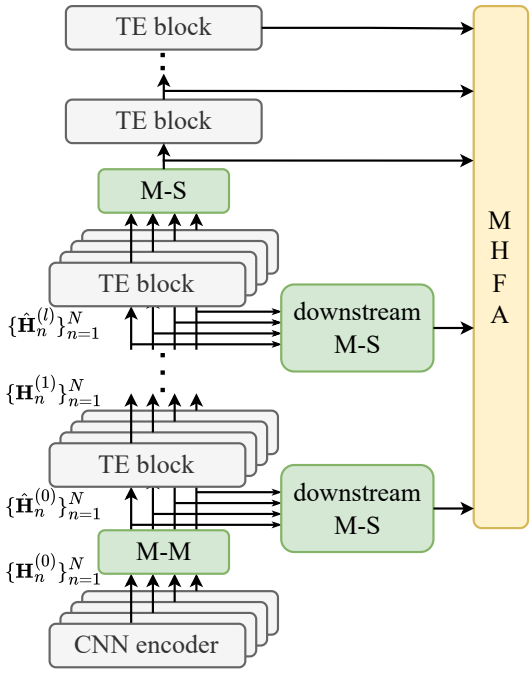



Figure 7.1: Designed multi-channel extension of SSL models (*METRO*) for speaker verification. TE stands for Transformer encoder, MHFA is multi-head factorized attentive pooling [Peng et al., 2023a].

thus applied to all channels due to weight sharing. As shown in [Luo et al., 2020], global information exchange provides clear benefits in speech enhancement/separation tasks. Following this idea, we append modules that promote cross-channel information fusion to the replicated original blocks. We introduce two types of such modules: *M-M* and *M-S*. While both have multi-channel input, M-M has a multi-channel output, and M-S provides a single-channel output. As displayed in Figure 7.1, M-M modules are inserted after each set of blocks until a specific layer. After the integration of the M-S module, the original single-channel structure follows. The resulting architecture resembles that of [Wang et al., 2020, Wang et al., 2021a], interleaving temporal per-channel processing with cross-channel processing. Our architecture, however, differs in that it builds upon the SSL model and eventually fuses parallel branches into a single one, which is computationally efficient and provides empirically corroborated performance improvements.

In order to perform speaker embedding extraction for SV, we employ a lightweight downstream model: multi-head factorized attentive pooling (MHFA). It was designed such that the outputs of the CNN encoder and the Transformer encoder blocks form the input to MHFA. In our multi-channel model, we propose to use M-M or M-S outputs instead. While the M-S outputs do not require special treatment, the M-M outputs must be compressed to a single channel via a *downstream M-S*. This approach allows for using the MHFA without modifications.

### 7.2.2 M-M Modules

The M-M modules allow information exchange between channels. We base their design on approaches from the literature [Luo et al., 2020, Horiguchi et al., 2022b]. We also

experimented with cross-channel attention [Wang et al., 2020] but found it inferior compared to the presented alternatives. As it will follow from the description, the utilized designs are agnostic to the number of channels.

Let $\mathbf{H}_n \in \mathbb{R}^{\mathfrak{T}\times\mathfrak{D}}$ be the output of the CNN or Transformer encoder block corresponding to the $n$th channel, where $\mathfrak{T}$ and $\mathfrak{D}$ represent the number of time frames (as extracted by the CNN encoder) and the feature dimensionality, respectively. Despite layer dependency, we omit the layer index $(\cdot)^{(l)}$ (see Figure 7.1) to ease notation. Let us define $f_{\text{FC}}(\mathbf{X}';\alpha) := \sigma(\mathbf{X}'\mathbf{W}+\mathbf{b})$, where $\alpha = \{\mathbf{W},\mathbf{b}\}$ and $\sigma(\cdot)$ is a suitable non-linearity. Moreover, let $f_{\text{MA}}(\mathbf{X},\mathbf{Y};\beta) := \left[\Big\|_{h=1}^{H} f_{\text{AT}}\left(\mathbf{X}\mathbf{W}_h^{(Q)}, \mathbf{X}\mathbf{W}_h^{(K)}, \mathbf{Y}\mathbf{W}_h^{(V)}\right)\right]\mathbf{W}_O$ be the $H$-head attention, where $f_{\text{AT}}(\mathbf{Q},\mathbf{K},\mathbf{V})$ is the dot-product attention defined in Eq. (1) of [Vaswani et al., 2017], the operator $||$ concatenates its arguments along the last dimension, and $\beta = \{\mathbf{W}_h^{(Q)}, \mathbf{W}_h^{(K)}, \mathbf{W}_h^{(V)}\}_{h=1}^{H} \cup \{\mathbf{W}_O\}$. The input matrix $\mathbf{X}$ defines queries and keys, hence attention weights.

### Transform-average-concatenate

The transform-average-concatenate (TAC) module presented in [Luo et al., 2020] averages frame-level per-channel representations into a single-channel representation in a $\bar{d}$-dimensional space parameterized by $\rho$. The resulting global representation

$$\mathbf{T} = f_{\text{FC}}\left(\frac{1}{N}\sum_{n=1}^{N} f_{\text{FC}}(\mathbf{H}_n;\rho);\chi\right) \tag{7.1}$$

is merged with the per-channel one as

$$\bar{\mathbf{T}}_n = \text{LN}\left(f_{\text{FC}}(\mathbf{H}_n||\mathbf{T};\varrho)\right), \tag{7.2}$$

with $\text{LN}(\cdot)$ denoting layer-norm. The output is then $\hat{\mathbf{H}}_n = \mathbf{H}_n + \bar{\mathbf{T}}_n$.

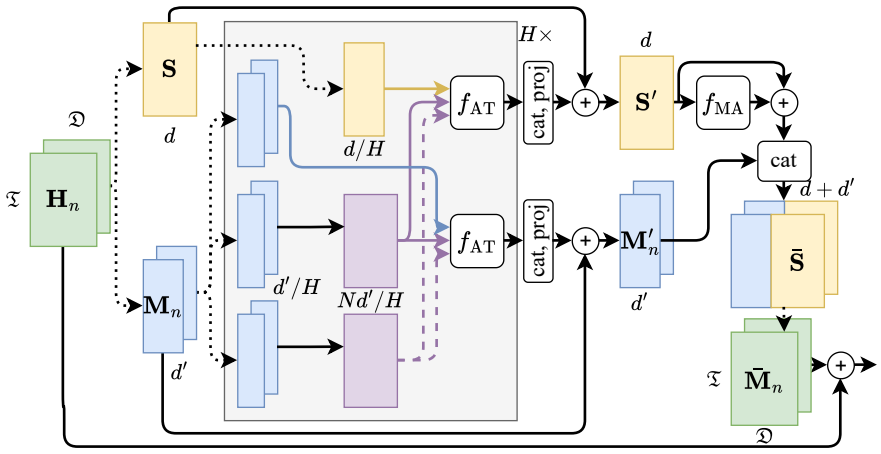



Figure 7.2: Scheme of the modified cross-frame co-attention. Dotted lines transform the data, "*cat*" is an abbreviation for concatenation, and "*proj*" for projection.

**Co-attention (COATT)**

Co-attention was introduced in the diarization field [Horiguchi et al., 2022b] to extend conventional end-to-end diarization (EEND-EDA [Horiguchi et al., 2022a]) to multiple channels. We modified the original co-attention module with two inputs and two outputs (single-channel summary and multi-channel representations) so that it can be used as the M-M module. Our version (Figure 7.2) performs similar high-level operations as TAC (summarization of channels, concatenation with per-channel representations, augmentation of the original channels). The difference is that both summary and per-channel representations are contextualized through cross-frame co-attention.

Given the multi-channel input $\{\mathbf{H}_n\}_{n=1}^{N}$, we obtain channels' summary embeddings as $\mathbf{S} = \mathrm{LN}\left(\left[\frac{1}{N}\sum_n \mathbf{H}_n\right]\mathbf{W}^{(S)}\right)$, where $\mathbf{W}^{(S)} \in \mathbb{R}^{\mathfrak{D}\times d}$ reduces dimensionality to $d$. Per-channel representations are given as $\mathbf{M}_n = \mathrm{LN}\left(\mathbf{H}_n\mathbf{W}^{(M)}\right) \in \mathbb{R}^{\mathfrak{T}\times d'}$. In the next step, cross-frame co-attention is performed. To this end, per-channel representations are concatenated: $\mathbf{M} = \mathop{\Big\|}_{n=1}^{N} \mathbf{M}_n \in \mathbb{R}^{\mathfrak{T}\times Nd'}$. The co-attention is then formulated as

$$\begin{aligned} \mathbf{M}'_n &= \mathrm{LN}\left(f_{\mathrm{MA}}(\mathbf{M}, \mathbf{M}_n; \vartheta) + \mathbf{M}_n\right), \\ \mathbf{S}' &= \mathrm{LN}\left(f_{\mathrm{MA}}(\mathbf{M}, \mathbf{S}; \vartheta') + \mathbf{S}\right). \end{aligned} \tag{7.3}$$

Both query $\mathcal{W}_h^{(Q)}$ and key $\mathcal{W}_h^{(K)}$ transformation matrices are shared by $\vartheta$ and $\vartheta'$ leading to the same attention matrix. Both $\mathcal{W}_h^{(Q)}$ and $\mathcal{W}_h^{(K)}$ are block-diagonal matrices composed of $N$ matrices $\mathbf{W}_h^{(Q)} \in \mathbb{R}^{d'\times d'/H}$ and $\mathbf{W}_h^{(K)} \in \mathbb{R}^{d'\times d'/H}$, respectively. To link the description and Figure 7.2, we note the equivalence $\mathbf{M}\mathcal{W}_h^{(Q)} = \mathop{\|}_{n}\mathbf{M}_n\mathbf{W}_h^{(Q)}$. Next, cross-frame self-attention is applied to the single-channel representation: $\bar{\mathbf{S}} = \mathrm{LN}\left(f_{\mathrm{MA}}(\mathbf{S}', \mathbf{S}'; \varpi) + \mathbf{S}'\right)$. Finally, $\bar{\mathbf{S}}$ is concatenated with each channel of $\mathbf{M}'_n$ and linear projection takes the result to a $\mathfrak{D}$-dimensional space: $\bar{\mathbf{M}}_n = \left(\mathbf{M}'_n || \bar{\mathbf{S}}\right)\mathbf{W}^{(F)}$, $\mathbf{W}^{(F)} \in \mathbb{R}^{(d+d')\times\mathfrak{D}}$. The output is $\hat{\mathbf{H}}_n = \mathbf{H}_n + \bar{\mathbf{M}}_n$.

### 7.2.3 M-S Modules

The M-S modules fuse multi-channel representations into single-channel ones. Since mean pooling is sufficient in the architecture with a similar high-level structure [Wang et al., 2020], we opted for a weighted average for the final M-S module. The channel weights are jointly optimized with the network. We found that they converge to the same values; hence, mean pooling suffices if the independence of the number of channels is of concern.

The downstream M-S modules are implemented either as a weighted average or the *take-first* approach. The latter simply uses the first channel (out of $N$) as its output.

## 7.3 Experiments

### 7.3.1 Experimental Setup

**Pre-trained model** Without loss of generality, we limit our experimentation to WavLM Base+ [Chen et al., 2022a]. The reason is twofold. First, WavLM-based models provide strong results on the SUPERB benchmark. Second, it represents a good trade-off between size and performance in SV [Peng et al., 2023a]. We note that our design is rather general

and is not restricted only to WavLM Base+. Larger models can potentially lead to better results.

**Training strategy** We take a two-stage training approach. In the first stage, the MHFA downstream model with 64 heads is appended to a single-channel pre-trained WavLM Base+. We follow the joint optimization scheme [Peng et al., 2023a], where the SSL model and MHFA are updated with learning rates of $2 \cdot 10^{-5}$ and $10^{-3}$, respectively. The model is trained on 3 s segments for 15 epochs, and the learning rates are decayed by a factor of 0.95 after each epoch. We use the additive angular margin loss (AAM, $s = 30$, $m = 0.2$) as an optimization objective [Deng et al., 2019]. We argue that access to a comprehensive multi-channel corpus could render this stage unnecessary. However, we empirically found the single-channel pre-training helpful.

In the second stage, we introduce the M-M and M-S into the architecture and jointly train the model on speaker-labeled multi-channel data with the same objective for 12 epochs. Newly incorporated weights are updated with a learning rate of $10^{-3}$, while others are updated with $2 \cdot 10^{-5}$. It follows from Figure 7.1 that the M-M modules in the backbone increasingly corrupt information flowing through the network if initialized randomly. We initialize the M-M modules to prevent the deeper layers from receiving polluted representations so that $\hat{\mathbf{H}}_n \approx \mathbf{H}_n$. In TAC, we achieve it by setting a multiplicative parameter built in layer-norm LN in (7.2) to $10^{-2}$. In COATT, values of $\mathbf{W}_F$ are drawn from $\mathcal{U}\left(-\sqrt{\frac{10^{-4}}{d+d'}}, \sqrt{\frac{10^{-4}}{d+d'}}\right)$. The final M-S and the downstream M-S are initialized to weigh channels equally.

By experimentation on the MultiSV dev retr set, we set the dimensions $\bar{d}$, $d$, and $d'$ to 960, 128, and 32, respectively. The number of heads $H$ is 8.

**Training data** In order to adapt the SSL model and MHFA to speaker recognition in the first training stage, we use VoxCeleb2 dev [Chung et al., 2018] augmented on the fly with MUSAN noise and reverberation. This dataset also serves for the baseline ResNet34 (with squeeze-and-excitation [Hu et al., 2020a]) embedding extractor training. The ResNet34 baseline model is the same one we used in Section 6.4.5.

The speakers in the multi-channel dataset must be a subset of those in the single-channel one to avoid architectural changes between stages. Such a requirement is satisfied by our MultiSV2 training data. At the same time, it provides a reasonably large set of examples for fine-tuning the whole model.

**Evaluation data** We use retransmitted read speech data of MultiSV for evaluation. Specifically, we follow the MRE and MRE_hard protocols that define verification trials with 4-channel enrollment and test parts. We recall that reverberation and noise are present in all enrollment and test segments except for the MRE enrollment (which is noise-free). We used the development subset to select hyperparameters, so the results on it are optimistic.

### 7.3.2 Towards the Multi-Channel Extension of the SSL Models

To put the embedding extractors into the context of SV, we first analyze the performance on VoxCeleb1-O. The ResNet34 extractor yields 1.08% EER and a minDCF of 0.086 (see Table 6.6). The first-stage-training single-channel extractor (*SC MHFA*) provides 0.96% EER and a minDCF of 0.129. In row A of Table 7.1, we show results on the MultiSV

Table 7.1: Experimental results on MultiSV retr. Results with parentheses are in the format *mean (std)*, where the moments were computed on four runs with different seeds. Downstream M-S is abbreviated as d. M-S. The final layer with the M-M module is layer 4. The metric minDCF is abbreviated as mDCF. The downstream M-S module with weighted average is abbreviated as *wavg. d. M-S*. We mark the results on the development part of MultSV in green since the results on this data were used to find hyperparameters.

| | Model | params. [M] | MRE dev. EER | MRE dev. mDCF | MRE eval. EER | MRE eval. mDCF | MRE_hard dev. EER | MRE_hard dev. mDCF | MRE_hard eval. EER | MRE_hard eval. mDCF |
|---|---|---|---|---|---|---|---|---|---|---|
| A | SC MHFA, rand. channel | 96.68 | 1.77 | 0.187 | 5.12 | 0.406 | 2.11 | 0.198 | 7.87 | 0.567 |
| B | FaSNet + SC MHFA | 99.44 | 1.95 | 0.181 | 6.69 | 0.447 | 2.27 | 0.188 | 10.44 | 0.644 |
| C | TasN-BF + SC MHFA | 97.88 | 1.33 | 0.135 | 3.03 | 0.293 | 1.31 | 0.138 | 3.72 | 0.334 |
| D | TasN-BF + ResNet34 | 19.66 | 1.00 | 0.090 | 2.88 | 0.274 | 1.19 | 0.095 | 3.62 | 0.332 |
| E | + joint fine-tuning (ft.) | 19.66 | 0.76 | 0.084 | 2.23 | 0.194 | 0.76 | 0.087 | 2.55 | **0.233** |
| F | SC MHFA embed. avg. | 96.68 | 1.07 | 0.125 | 2.48 | 0.277 | 1.40 | 0.134 | 4.57 | 0.430 |
| G | SC MHFA parallel fixed | 96.68 | 1.44 | 0.119 | 2.16 | 0.228 | 1.55 | 0.125 | 3.15 | 0.342 |
| H | SC MHFA parallel trained | 96.68 | 0.90 (0.04) | 0.099 | 1.82 (0.06) | 0.184 | 1.11 (0.02) | 0.108 | 2.30 (0.06) | 0.283 |
| I | METRO w/ TAC | 112.37 | 0.77 (0.03) | 0.096 | 1.52 (0.03) | 0.187 | 1.01 (0.06) | 0.108 | 1.85 (0.11) | 0.272 |
| J | + wavg. d. M-S | 112.37 | 0.84 (0.04) | 0.087 | 1.57 (0.06) | 0.178 | 1.09 (0.03) | 0.102 | 1.87 (0.04) | 0.263 |
| K | METRO w/ COATT | 98.44 | 0.80 (0.02) | 0.095 | 1.65 (0.07) | 0.181 | 1.02 (0.04) | 0.107 | 1.85 (0.02) | 0.269 |
| L | + wavg. d. M-S | 98.44 | 0.76 (0.07) | 0.084 | **1.50** (0.04) | **0.162** | 0.94 (0.07) | 0.095 | **1.79** (0.06) | 0.240 |

retransmitted data when a random channel per microphone array is selected and the embedding is extracted with SC MHFA. It can be seen as a baseline for systems building on top of it.

Next, we present intuitive ways to extend SSL models to multi-channel data and justify the benefits of information fusion and, thus, our design. Following [Cai and Li, 2021], a straightforward approach is to extract per-channel embeddings with SC MHFA and average them (row F of Table 7.1). Despite its simplicity, it considerably boosts performance. The downside is that the model was trained on VoxCeleb2 dev. Therefore, a domain shift between training and evaluation data exists. To provide the ability to adapt to multi-channel data, we devise another baseline (row G). The SSL model processes channels in parallel, and downstream M-S, implemented by weighted average, provides input to the MHFA (the only updated part). This baseline provides clear benefits, especially for the evaluation sets of MRE and MRE_hard. In row H, we present the performance of a strong baseline — a model with the same architecture where we allow fine-tuning of the SSL model (using learning rates according to Section 7.3.1).

Since we devoted much attention to multi-channel pre-processing in the previous chapters, we also compare the abovementioned baselines with cascaded models. First, we present an alternative system to model B in Table 6.8. The difference compared to model B is that the SC MHFA replaced the original ResNet34-based embedding extractor. The multi-channel pre-processing is implemented by FaSNet TAC [Luo et al., 2020] trained on the training part of MultiSV. Although it provides outputs with audibly reduced distractors, distortions introduced by the network result in a domain shift, degrading the performance compared to SC MHFA (row B). Row D represents a cascaded model with Conv-TasNet separator, indirect mask prediction for mask-based MVDR beamforming, and ResNet34 embedding extractor (i.e., model D in Table 6.8). It tends to outperform row F, which was also not fine-tuned on multi-channel data. Row C is a version of the row D model, replacing the embedding extractor with SC MHFA. Although it performs somewhat worse, it is interesting to compare C with B. In that case, the embedding extractor is the same, but the pre-processing differs. Confirming our previous findings in Chapter 6, the beamforming-based model considerably outperforms that using the neural network's direct output. A

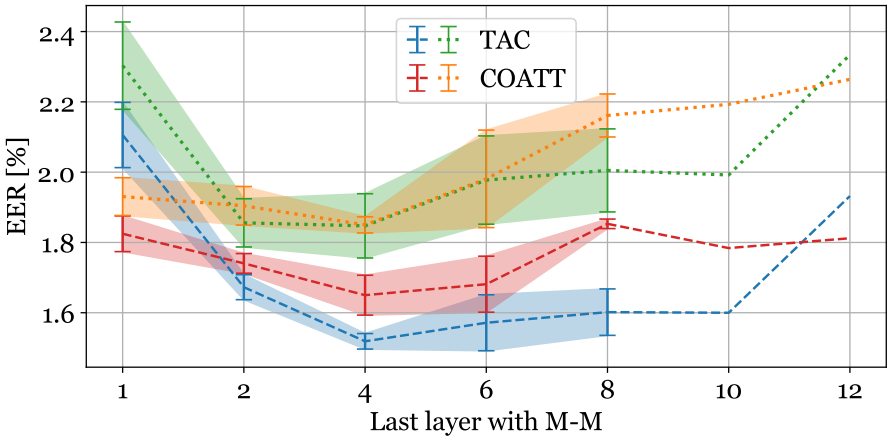



Figure 7.3: Performance dependence on the last layer where the M-M module is inserted (with the take-first downstream M-S). Red and blue colors correspond to the MRE protocol, green and orange to the MRE_hard protocol. Shaded areas depict standard deviations obtained on four runs.

version of D fine-tuned on multi-channel data is in row E. This model corresponds to model D.3 in Table 6.9.

Rows I–L present the results achieved with the METRO models proposed in this chapter. The M-M modules were incorporated according to Figure 7.1, where the last layer with M-M was selected based on Section 7.3.3. In rows I and K, the downstream M-S modules follow the take-first approach. The weighted average approach to the downstream M-S provides results in rows J and L. With the take-first approach to downstream M-S, TAC and COATT perform on par on MRE_hard, while COATT requires many fewer parameters. Downstream M-S implemented by weighted average improves only minDCF in METRO with TAC. However, COATT consistently benefits from it, providing the best results overall. It is worth noting that the baseline H is more computationally expensive than any METRO instance as all channels are forwarded through the whole backbone.

### 7.3.3 Layer Dependence

We empirically found that keeping parallel processing throughout the whole architecture is not necessarily optimal. We hypothesize that keeping parallel processing at the beginning is beneficial because representations that are closer to the signals provide complementary and spatial information. The fusion of abstract representations extracted further within the model does not provide improvements.

In Figure 7.3, we present ablation results where we insert the M-M modules up to the layer denoted on the x-axis. While the decision on the last (4th) layer, where the M-M is inserted in our final models, was made on the development part of MultiSV, we note that trends are similar on the development (which we do not show) and evaluation subsets.

## 7.4 Summary

This chapter presented METRO, a Multi-channel ExTension of pRe-trained mOdels for SV, outperforming previously presented and published cascaded models on MultiSV. It replicates the original blocks of the SSL models to perform parallel processing. They are

followed by modules providing channel information exchange. Channels are eventually fused into one, and single-channel processing follows. Given per-layer representations, we used MHFA to extract speaker embeddings.

Even though we experimented with WavLM Base+, our approach is general. Therefore, extension to other architectures is straightforward, and gains from large ones are expected. In this chapter, we presented METRO in the context of SV. However, exploration for other tasks (such as ASR) is an interesting research direction.

# Chapter 8

# Conclusions

This thesis was concerned with issues encountered in far-field speaker verification. We especially focused on the microphone arrays as a means to alleviate them. We concentrated on two fundamental aspects of fostering research in multi-channel speaker verification: data and models leveraging multiple microphones.

The scarcity and the lack of a common data hinder new development and comparison of various techniques. Our goal in this direction was to create a publicly available corpus that can be easily adopted. At the beginning of our endeavors, we contributed to the BUT ReverbDB dataset [Szőke et al., 2019], which (among others) provides multi-channel evaluation data for speaker verification. The limitation of it is the license. Since the retransmitted recordings are based on the data from NIST evaluations, a valid LDC license is required to obtain a copy. Our efforts related to the data problem resulted in designing a publicly available MultiSV dataset. It provides multi-channel training data containing mixtures of speech and noise. The data comes with the signals of sources in the mixtures (for front-end training) and speaker labels (for embedding extractors). MultiSV also defines multiple evaluation protocols based on retransmitted data, supporting various use cases (single-/multi-channel enrollment and multi-channel test), and corresponding development data. Finally, it contains simulated development and evaluation data for front-end analysis (due to providing mixture sources). The downside of the training data is its size. It contains only about 73 hours of audio per microphone. Therefore, it is not suitable for training embedding extractors from scratch. Attempting to tackle this issue, we released MultiSV2 training data containing approximately 1,160 hours of audio per microphone[1]. Although it is a significant increase, MultiSV2 still lags behind recent single-channel datasets (such as VoxBlink2), which have been remarkably growing in size even for use in academia. Another weakness of the MultiSV2 (and MultiSV) training data is the data simulation, which only approximates real acoustics and causes a mismatch regarding the evaluation data.

Concerning developing far-field SV systems, our research question was how to build models dealing with speech recorded in room conditions with limited training data. Such evaluation speech recordings are commonly degraded by reverberation and noise. We started by exploring the multiplicity of microphones. We confirmed in many experiments that multi-channel processing can lead to substantial improvements over using just one far-field microphone. Even though it may seem obsolete, we investigated beamforming for its benefits. It helped us decompose the problem at hand into multi-channel processing and single-channel

[1]We note that noises from the CHiME-3 challenge require an LDC license. They were selected to enhance the diversity. However, the CHiME-3 noises are the least represented ones in the collection. Therefore, replacing them with other noises we selected for MultiSV2 should not have a significant impact.

Table 8.1: Summary of selected models tracking the achieved progress in multi-channel SV. The presented models correspond to the following ones in this thesis: I) the last model in Table 6.3, II, III) the first and sencond models in Table 6.5, IV, V) models J and N in Table 6.8, VI) model N.3 in Table 6.9, and VII) model L in Table 7.1. MinDCF is abbreviated as mDCF.

| | Model | params. [M] | MRE eval. EER | MRE eval. mDCF | MRE_hard eval. EER | MRE_hard eval. mDCF | SRE EER | SRE mDCF |
|---|---|---|---|---|---|---|---|---|
| | Diff-Filter [Dowerah et al., 2023a] | — | 3.07 | — | 3.19 | — | — | — |
| I | Application aware [Mošner et al., 2019] | — | — | — | — | — | 4.02 | 0.427 |
| II | Mask predictor [Mošner et al., 2022b] | 17.16 | 3.91 | 0.355 | 5.37 | 0.518 | 4.47 | 0.354 |
| III | Conv-TasNet-based [Mošner et al., 2022b] | 15.16 | 3.71 | 0.364 | 4.61 | 0.482 | 4.39 | 0.346 |
| IV | DPTNet-based [Mošner et al., 2025] | 17.43 | 2.62 | 0.256 | 3.16 | 0.295 | **2.56** | 0.211 |
| V | DPTNet+RCA [Mošner et al., 2025] | 17.55 | 2.50 | 0.250 | 3.01 | 0.283 | **2.56** | **0.209** |
| VI | DPTNet+RCA finetuned [Mošner et al., 2025] | 17.55 | 2.21 | 0.189 | 2.44 | **0.226** | — | — |
| VII | METRO [Mošner et al., 2024] | 98.44 | **1.50** | **0.162** | **1.79** | 0.240 | — | — |

embedding extraction, where the former is typically less data-demanding. Moreover, the employed beamforming can be interpreted as a filter-and-sum operation (summing linearly filtered channel signals). The benefit of it is that it does not distort the data in a way common to direct multi-channel neural network-based pre-processors. In multiple experiments, we confirmed that using neural network multi-channel models providing inputs to the embedding extractor in data-limited applications without retraining the extractor on the processed data is suboptimal and degrades results compared to beamforming. Affirming that cascaded multi-channel SV is viable, we were interested in quantifying the impact of sole pre-processing on the SV performance. We observed that a significant improvement in separation leads to better estimates of SCMs and improved beamforming. Better beamforming, in turn, indeed leads to boosting SV performance. However, limited improvements in separation lead to comparably performing beamforming, not altering the final SV metrics. This observation motivated us to explore multi-channel speech-noise separation. The exploration provided various insights: for instance, we observed how the separation (and SV) performance improves with the increase in the window size of STFT. We confirmed a known fact about improvements stemming from the increase in microphone count. We showed the contradicting effect of mask-based and maskless beamforming on separation and verification results. We also presented how extending speech separation models to multiple channels via RCA can benefit both beamforming separation performance and, more importantly and significantly, the final SV performance.

Although the cascaded approach conveniently decomposes our problem into two tasks, it has limitations. Out of the fundamental ones, both subsystems first optimize different

(not necessarily well-correlated) criteria. Another issue is that errors from the front-end propagate to the embedding extractor. We initially attempted to alleviate the issue by fine-tuning the joint models' parameters with a speaker classification loss. In the data-limited regime, we updated only the front-end. Employing MultiSV2, we devised the approach that eventually allowed us to update all the parameters. We confirmed that both approaches helped update parameters in a way that was beneficial for our final task.

Although fine-tuning of the joint cascaded models confirmed our hypothesis on the benefits, there are still issues. Obtaining the final fine-tuned model is a complex task with multiple steps and different data requirements. Therefore, our final goal was to develop a holistic model that would not rely on decomposing the problem into multi-channel separation and embedding extraction. To this end, we designed a technique extending foundation speech models to multi-channel ones suitable for embedding extraction. This approach was allowed by the MultiSV2 training data. It eventually managed to outperform variants of cascaded models by a large margin.

In Table 8.1, we summarize our designs. They reflect the progress we achieved with different variants of cascaded models and, finally, with the approach employing pre-trained SSL models.

## 8.1 Future Work

Although we provided the community with the dataset, potentially allowing for comparison and competition, it has limitations. One of the issues is that the training data is simulated; therefore, it only loosely approximates real-world conditions. Our training datasets, MultiSV and MultiSV2, assume that the source of distraction is active for the whole duration of the target speaker's speech. Although this might be the case for some noises (such as a fan or AC), it is definitely not true for all cases (door closing, dog barking, etc.). The dataset is also limited to employing only four-channel arrays. A common issue of both training and evaluation data is the fact that the positions of sources are static. In real-world applications, users and/or noise sources will likely move during the recording session. The outlined limitations provide directions for new iterations of multi-channel SV verification datasets. Last but not least, with the ever-increasing size of models, demands on data increase as well. Hence, scalability plays an important role in data preparation.

In the cascaded systems, we reflected the speech separation/enhancement progress. The evolution in recent years has brought new and more powerful models, such as TF-GridNet [Wang et al., 2023a]. Since we observed that improved separation results in improved SV performance, it would be worthwhile exploring the limits of the cascaded systems in light of new trends in separation.

We relied on well-defined beamformers (MVDR and GEV) in our cascaded systems. Although they are theoretically sound and interpretable, the fact that they result from optimization of specific criteria upperbounds their performance. Therefore, one potential line of research could consider less explicit beamforming, representing an intermediate step between cascaded systems and fully end-to-end models. Previously discussed all-neural beamforming and its versions are candidates for such an implicit beamformer.

In our experiments with front-ends, we often observed relatively low mean SDRi values. It was commonly caused by low SDRi for the noise components. While source dominance is usually not obvious in speaker separation, the speaker source dominates over the noise source in our data, which stems from the use case. In turn, errors in the target speaker prediction degrade the loss more during speech separation model training. Therefore, the

networks learn to provide a better quality of separated speech than separated noise. This optimization issue is worth reconsidering since improving the noise estimate is of interest, as it participates in the beamforming vector prediction.

In our METRO approach, we made use of existing single-channel foundation models. Some studies reported a beneficial effect of unsupervised training on in-domain data (even with smaller volumes of data compared to traditional foundation models' training data, e.g., [Peng et al., 2023b]). Therefore, an interesting area to explore is the pre-training of the backbone model in the SSL manner. One possible way would be to perform pre-training on single-channel data coming from a similar domain (far-field data). Arguably, there are more resources for single-channel far-field data compared to multi-channel data. Then the extension could follow our initial METRO proposal. Another appealing way is pre-training on multi-channel data. In such a case, we could allow ourselves to explore architectural changes more freely.

A weakness of the METRO-style multi-channel SV model (apparent in Table 8.1) is its size, which can be limiting. It is an inherent property due to using pre-trained models. In order to keep using SSL models and benefit from a smaller final size, a suitable pruning method could be applied. One such technique that was proven helpful is structured pruning [Louizos et al., 2018, Han et al., 2025]. The benefits obtained using SSL models to build the embedding extractor without signal-level pre-processing paved the way to new holistic models. While there is a lot of potential, this field is yet to be explored.

In general, multi-channel SV is still a niche field with a limited number of publications. It is questionable how it will evolve in the era emphasizing more complex tasks, such as speaker-attribute automatic speech systems, multi-modal and multi-lingual models, and large language models in speech processing. Nevertheless, we believe that multi-channel speaker embedding extraction and outcomes of our work related to it may be relevant in the future. Speaker embeddings are not employed only in SV but are valuable for other tasks, such as speaker extraction, target speaker ASR, or voice conversion.

# Bibliography


[Allen and Berkley, 1979] Allen, J. and Berkley, D. (1979). Image method for efficiently simulating small-room acoustics. *Journal of the Acoustical Society of America*, 65(4):943–950.

[Anguera et al., 2007] Anguera, X., Wooters, C., and Hernando, J. (2007). Acoustic beamforming for speaker diarization of meetings. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(7):2011–2021.

[Arandjelovic et al., 2016] Arandjelovic, R., Gronat, P., Torii, A., Pajdla, T., and Sivic, J. (2016). NetVLAD: CNN architecture for weakly supervised place recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5297–5307.

[Ba et al., 2016] Ba, J. L., Kiros, J. R., and Hinton, G. E. (2016). Layer normalization. *arXiv preprint*. arXiv:1607.06450 [stat.ML].

[Baevski et al., 2020] Baevski, A., Zhou, Y., Mohamed, A., and Auli, M. (2020). wav2vec 2.0: A framework for self-supervised learning of speech representations. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 12449–12460. Curran Associates, Inc.

[Barker et al., 2015] Barker, J., Marxer, R., Vincent, E., and Watanabe, S. (2015). The third 'CHiME' speech separation and recognition challenge: Dataset, task and baselines. In *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 504–511.

[Barker et al., 2018] Barker, J., Watanabe, S., Vincent, E., and Trmal, J. (2018). The fifth 'CHiME' speech separation and recognition challenge: Dataset, task and baselines. In *Proc. Interspeech*, pages 1561–1565.

[Beltagy et al., 2020] Beltagy, I., Peters, M. E., and Cohan, A. (2020). Longformer: The long-document transformer. *arXiv preprint*. arXiv:2004.05150 [cs.CL].

[Benesty et al., 2008] Benesty, J., Chen, J., and Huang, Y. (2008). *Microphone Array Signal Processing*. Springer Topics in Signal Processing. Springer Berlin Heidelberg. ISBN: 9783540786122.

[Bhattacharya et al., 2016] Bhattacharya, G., Alam, J., Kenn, P., and Gupta, V. (2016). Modelling speaker and channel variability using deep neural networks for robust speaker verification. In *2016 IEEE Spoken Language Technology Workshop (SLT)*, pages 192–198.

[Boeddeker et al., 2017] Boeddeker, C., Hanebrink, P., Drude, L., Heymann, J., and Haeb-Umbach, R. (2017). Optimizing neural-network supported acoustic beamforming by algorithmic differentiation. In *ICASSP 2017 - 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 171–175.

[Brown et al., 2022] Brown, A., Huh, J., Chung, J. S., Nagrani, A., and Zisserman, A. (2022). VoxSRC 2021: The third VoxCeleb speaker recognition challenge. *arXiv preprint*. arXiv:2201.04583 [cs.SD].

[Brümmer, 2010a] Brümmer, N. (2010a). EM for probabilistic LDA (PLDA). Technical report, Agnitio Research, South Africa.

[Brümmer, 2010b] Brümmer, N. (2010b). EM for simple PLDA. Technical report, Agnitio Research, South Africa.

[Brümmer, 2010c] Brümmer, N. (2010c). *Measuring, Refining and Calibrating Speaker and Language Information Extracted From Speech*. Ph.D. thesis, Stellenbosch: University of Stellenbosch.

[Brümmer and Doddington, 2013] Brümmer, N. and Doddington, G. R. (2013). Likelihood-ratio calibration using prior-weighted proper scoring rules. In *Proc. Interspeech*, pages 1976–1980.

[Brümmer et al., 2018] Brümmer, N., Silnova, A., Burget, L., and Stafylakis, T. (2018). Gaussian meta-embeddings for efficient scoring of a heavy-tailed PLDA model. In *The Speaker and Language Recognition Workshop (Odyssey 2018)*, pages 349–356.

[Brümmer et al., 2022] Brümmer, N., Swart, A., Mošner, L., Silnova, A., Plchot, O., Stafylakis, T., and Burget, L. (2022). Probabilistic spherical discriminant analysis: An alternative to PLDA for length-normalized embeddings. In *Proc. Interspeech*, pages 1446–1450.

[Burget et al., 2008] Burget, L., Brümmer, N., Reynolds, D., Kenny, P., Pelecanos, J. W., Vogt, R., Castaldo, F., Dehak, N., Dehak, R., Glembek, O., Karam, Z. N., Noecker, J., Costin, C., Hubeika, V., Kajarekar, S. S., and Scheffer, N. (2008). Robust speaker recognition over varying channels report from JHU workshop 2008. Technical report, Johns Hopkins University.

[Burget et al., 2011] Burget, L., Plchot, O., Cumani, S., Glembek, O., Matějka, P., and Brümmer, N. (2011). Discriminatively trained probabilistic linear discriminant analysis for speaker verification. In *ICASSP 2011 - 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4832–4835.

[Cai and Li, 2021] Cai, D. and Li, M. (2021). Embedding aggregation for far-field speaker verification with distributed microphone arrays. In *2021 IEEE Spoken Language Technology Workshop (SLT)*, pages 308–315.

[Cai et al., 2019] Cai, D., Qin, X., and Li, M. (2019). Multi-channel training for end-to-end speaker recognition under reverberant and noisy environment. In *Proc. Interspeech*, pages 4365–4369.

[Cano et al., 2016] Cano, E., FitzGerald, D., and Brandenburg, K. (2016). Evaluation of quality of sound source separation algorithms: Human perception vs quantitative metrics. In *2016 24th European Signal Processing Conference (EUSIPCO)*, pages 1758–1762.

[Capon, 1969] Capon, J. (1969). High-resolution frequency-wavenumber spectrum analysis. *Proceedings of the IEEE*, 57(8):1408–1418.

[Chang et al., 2021] Chang, F.-J., Radfar, M., Mouchtaris, A., King, B., and Kunzmann, S. (2021). End-to-end multi-channel transformer for speech recognition. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5884–5888.

[Chen et al., 2020] Chen, J., Mao, Q., and Liu, D. (2020). Dual-path transformer network: Direct context-aware modeling for end-to-end monaural speech separation. In *Proc. Interspeech*, pages 2642–2646.

[Chen et al., 2022a] Chen, S., Wang, C., Chen, Z., Wu, Y., Liu, S., Chen, Z., Li, J., Kanda, N., et al. (2022a). WavLM: Large-scale self-supervised pre-training for full stack speech processing. *IEEE Journal of Selected Topics in Signal Processing*, 16(6):1505–1518.

[Chen et al., 2022b] Chen, Z., Chen, S., Wu, Y., Qian, Y., Wang, C., Liu, S., Qian, Y., and Zeng, M. (2022b). Large-scale self-supervised speech representation learning for automatic speaker verification. In *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6147–6151.

[Chen et al., 2023] Chen, Z., Han, B., Xiang, X., Huang, H., Liu, B., and Qian, Y. (2023). Build a SRE challenge system: Lessons from VoxSRC 2022 and CNSRC 2022. In *Proc. Interspeech*, pages 3202–3206.

[Choi et al., 2019] Choi, H.-S., Kim, J., Huh, J., Kim, A., Ha, J.-W., and Lee, K. (2019). Phase-aware speech enhancement with deep complex U-Net. In *International Conference on Learning Representations*.

[Chung et al., 2019] Chung, J. S., Nagrani, A., Coto, E., Xie, W., McLaren, M., Reynolds, D. A., and Zisserman, A. (2019). VoxSRC 2019: The first VoxCeleb speaker recognition challenge. *arXiv preprint*. arXiv:1912.02522 [cs.SD].

[Chung et al., 2018] Chung, J. S., Nagrani, A., and Zisserman, A. (2018). VoxCeleb2: Deep speaker recognition. In *Proc. Interspeech*, pages 1086–1090.

[Compernolle et al., 1989] Compernolle, D. V., Ma, W., and Diest, M. V. (1989). Speech recognition in noisy environments with the aid of microphone arrays. In *Proc. Eurospeech*, pages 2657–2660.

[Cornell et al., 2023] Cornell, S., Wiesner, M. S., Watanabe, S., Raj, D., Chang, X., Garcia, P., Masuyam, Y., Wang, Z.-Q., et al. (2023). The CHiME-7 DASR challenge: Distant meeting transcription with multiple devices in diverse scenarios. In *Proc. 7th International Workshop on Speech Processing in Everyday Environments (CHiME 2023)*, pages 1–6.

[Cosentino et al., 2020] Cosentino, J., Pariente, M., Cornell, S., Deleforge, A., and Vincent, E. (2020). LibriMix: An open-source dataset for generalizable speech separation. *arXiv preprint*. arXiv:2005.11262 [eess.AS].

[Cumani and Laface, 2014a] Cumani, S. and Laface, P. (2014a). Large-scale training of pairwise support vector machines for speaker recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 22(11):1590–1600.

[Cumani and Laface, 2014b] Cumani, S. and Laface, P. (2014b). Training pairwise support vector machines with large scale datasets. In *ICASSP 2014 - 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1645–1649.

[Defferrard et al., 2017] Defferrard, M., Benzi, K., Vandergheynst, P., and Bresson, X. (2017). FMA: A dataset for music analysis. In *ISMIR*. arXiv:1612.01840 [cs.SD].

[Dehak et al., 2011] Dehak, N., Kenny, P. J., Dehak, R., Dumouchel, P., and Ouellet, P. (2011). Front-end factor analysis for speaker verification. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(4):788–798.

[Deng et al., 2019] Deng, J., Guo, J., Xue, N., and Zafeiriou, S. (2019). ArcFace: Additive angular margin loss for deep face recognition. In *2019 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4690–4699.

[Desplanques et al., 2020] Desplanques, B., Thienpondt, J., and Demuynck, K. (2020). ECAPA-TDNN: Emphasized channel attention, propagation and aggregation in TDNN based speaker verification. In *Proc. Interspeech*, pages 3830–3834.

[Diaz-Guerra et al., 2021] Diaz-Guerra, D., Miguel, A., and Beltran, J. R. (2021). gpuRIR: A python library for room impulse response simulation with GPU acceleration. *Multimedia Tools and Applications*, 80(4):5653–5671.

[Dimitriadis et al., 2023] Dimitriadis, A., Pan, S., Sethu, V., and Ahmed, B. (2023). Spatial HuBERT: Self-supervised spatial speech representation learning for a single talker from multi-channel audio. *arXiv preprint*. arXiv:2310.10922 [cs.CL].

[Doddington, 1998] Doddington, G. R. (1998). Speaker recognition evaluation methodology: a review and perspective. In *RLA2C Workshop: Speaker Recognition and its Commercial and Forensic Applications*, page 60–66.

[Dowerah et al., 2023a] Dowerah, S., Kulkarni, A., Serizel, R., and Jouvet, D. (2023a). Self-supervised learning with diffusion-based multichannel speech enhancement for speaker verification under noisy conditions. In *Proc. Interspeech*, pages 3849–3853.

[Dowerah et al., 2023b] Dowerah, S., Serizel, R., Jouvet, D., Mohammadamini, M., and Matrouf, D. (2023b). Joint optimization of diffusion probabilistic-based multichannel speech enhancement with far-field speaker verification. In *2022 IEEE Spoken Language Technology Workshop (SLT)*, pages 428–435.

[Défossez et al., 2020] Défossez, A., Synnaeve, G., and Adi, Y. (2020). Real time speech enhancement in the waveform domain. In *Proc. Interspeech*, pages 3291–3295.

[Erdogan et al., 2015] Erdogan, H., Hershey, J. R., Watanabe, S., and Le Roux, J. (2015). Phase-sensitive and recognition-boosted speech separation using deep recurrent neural networks. In *ICASSP 2015 - 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 708–712.

[Erdogan et al., 2016] Erdogan, H., Hershey, J. R., Watanabe, S., Mandel, M. I., and Roux, J. L. (2016). Improved MVDR beamforming using single-channel mask prediction networks. In *Proc. Interspeech*, pages 1981–1985.

[Fan et al., 2021] Fan, Z., Li, M., Zhou, S., and Xu, B. (2021). Exploring wav2vec 2.0 on speaker verification and language identification. In *Proc. Interspeech*, pages 1509–1513.

[Fazel-Zarandi and Hsu, 2023] Fazel-Zarandi, M. and Hsu, W.-N. (2023). Cocktail HuBERT: Generalized self-supervised pre-training for mixture and single-source speech. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5.

[Ferrer et al., 2011] Ferrer, L., Bratt., H., Burget, L., Černocký, J., Glembek, O., Graciarena, M., Lawson, A., Lei, Y., Matějka, P., Plchot, O., and Scheffer, N. (2011). Promoting robustness for speaker modeling in the community: the PRISM evaluation set. In *Proceedings of SRE11 Analysis Workshop in 2011*, pages 1–7, Atlanta, Georga.

[Fonseca et al., 2022] Fonseca, E., Favory, X., Pons, J., Font, F., and Serra, X. (2022). FSD50K: An open dataset of human-labeled sound events. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:829–852.

[Fu et al., 2017a] Fu, S.-W., Hu, T.-y., Tsao, Y., and Lu, X. (2017a). Complex spectrogram enhancement by convolutional neural network with multi-metrics learning. In *2017 IEEE 27th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6.

[Fu et al., 2018] Fu, S.-W., Tsao, Y., Hwang, H.-T., and Wang, H.-M. (2018). Quality-Net: An end-to-end non-intrusive speech quality assessment model based on BLSTM. In *Proc. Interspeech*, pages 1873–1877.

[Fu et al., 2017b] Fu, S.-W., Tsao, Y., Lu, X., and Kawai, H. (2017b). Raw waveform-based speech enhancement by fully convolutional networks. In *2017 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 6–12.

[Fukushima, 1969] Fukushima, K. (1969). Visual feature extraction by a multilayered network of analog threshold elements. *IEEE Transactions on Systems Science and Cybernetics*, 5(4):322–333.

[Gao et al., 2021] Gao, S.-H., Cheng, M.-M., Zhao, K., Zhang, X.-Y., Yang, M.-H., and Torr, P. (2021). Res2Net: A new multi-scale backbone architecture. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(2):652–662.

[Garcia-Romero et al., 2019] Garcia-Romero, D., Snyder, D., Watanabe, S., Sell, G., McCree, A., Povey, D., and Khudanpur, S. (2019). Speaker recognition benchmark using the CHiME-5 corpus. In *Proc. Interspeech*, pages 1506–1510.

[Glembek, 2012] Glembek, O. (2012). *Optimization of Gaussian Mixture Subspace Models and Related Scoring Algorithms in Speaker Verification*. Ph.D. thesis, Brno University of Technology, Brno, Czechia.

[Glembek et al., 2011] Glembek, O., Burget, L., Brümmer, N., Plchot, O., and Matějka, P. (2011). Discriminatively trained i-vector extractor for speaker verification. In *Proc. Interspeech*, pages 137–140.

[Glembek et al., 2014] Glembek, O., Ma, J., Matějka, P., Zhang, B., Plchot, O., Burget, L., and Matsoukas, S. (2014). Domain adaptation via within-class covariance correction in i-vector based speaker recognition systems. In *ICASSP 2014 - 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4060–4064.

[Gonzalez-Rodriguez et al., 1996] Gonzalez-Rodriguez, J., Ortega-Garcia, J., Martin, C., and Hernandez, L. (1996). Increasing robustness in GMM speaker recognition systems for noisy and reverberant speech with low complexity microphone arrays. In *Proceeding of Fourth International Conference on Spoken Language Processing. ICSLP '96*, volume 3, pages 1333–1336.

[Griffiths and Jim, 1982] Griffiths, L. and Jim, C. (1982). An alternative approach to linearly constrained adaptive beamforming. *IEEE Transactions on Antennas and Propagation*, 30(1):27–34.

[Gu et al., 2019] Gu, R., Wu, J., Zhang, S., Chen, L., Xu, Y., Yu, M., Su, D., Zou, Y., and Yu, D. (2019). End-to-end multi-channel speech separation. *arXiv preprint*. arXiv:1905.06286 [cs.SD].

[Gu et al., 2023] Gu, R., Zhang, S.-X., Zou, Y., and Yu, D. (2023). Towards unified all-neural beamforming for time and frequency domain speech separation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31:849–862.

[Guan et al., 2021] Guan, S., Liu, S., Chen, J., Zhu, W., Li, S., Tan, X., Yang, Z., Xu, M., Chen, Y., Wang, J., and Zhang, X.-L. (2021). Libri-adhoc40: A dataset collected from synchronized ad-hoc microphone arrays. *arXiv preprint*. arXiv:2103.15118 [eess.AS].

[Han et al., 2025] Han, J., Landini, F., Rohdin, J., Silnova, A., Diez, M., Černocký, J., and Burget, L. (2025). Fine-tune before structured pruning: Towards compact and accurate self-supervised models for speaker diarization. In *Proc. Interspeech*, pages 1583–1587.

[Hao et al., 2021] Hao, X., Su, X., Horaud, R., and Li, X. (2021). FullSubNet: A full-band and sub-band fusion model for real-time single-channel speech enhancement. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6633–6637.

[He et al., 2016a] He, K., Zhang, X., Ren, S., and Sun, J. (2016a). Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

[He et al., 2016b] He, K., Zhang, X., Ren, S., and Sun, J. (2016b). Identity mappings in deep residual networks. In Leibe, B., Matas, J., Sebe, N., and Welling, M., editors,

*Computer Vision – ECCV 2016*, pages 630–645, Cham. Springer International Publishing.

[Heigold et al., 2016] Heigold, G., Moreno, I., Bengio, S., and Shazeer, N. (2016). End-to-end text-dependent speaker verification. In *ICASSP 2016 - 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5115–5119.

[Heitkaemper et al., 2020] Heitkaemper, J., Jakobeit, D., Boeddeker, C., Drude, L., and Haeb-Umbach, R. (2020). Demystifying TasNet: A dissecting approach. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6359–6363.

[Hershey et al., 2016] Hershey, J. R., Chen, Z., Le Roux, J., and Watanabe, S. (2016). Deep clustering: Discriminative embeddings for segmentation and separation. In *ICASSP 2016 - 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 31–35.

[Heymann et al., 2017] Heymann, J., Drude, L., Boeddeker, C., Hanebrink, P., and Haeb-Umbach, R. (2017). Beamnet: End-to-end training of a beamformer-supported multi-channel ASR system. In *ICASSP 2017 - 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5325–5329.

[Heymann et al., 2016a] Heymann, J., Drude, L., and Haeb-Umbach, R. (2016a). Neural network based spectral mask estimation for acoustic beamforming. In *ICASSP 2016 - 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 196–200.

[Heymann et al., 2016b] Heymann, J., Drude, L., and Haeb-Umbach, R. (2016b). Wide residual BLSTM network with discriminative speaker adaptation for robust speech recognition. In *Proceedings of the 4th International Workshop on Speech Processing in Everyday Environments (CHiME'16)*.

[Heymann et al., 2017] Heymann, J., Drude, L., and Haeb-Umbach, R. (2017). A generic neural acoustic beamforming architecture for robust multi-channel speech processing. *Computer Speech & Language*, 46:374–385.

[Higuchi et al., 2017] Higuchi, T., Kinoshita, K., Delcroix, M., Žmolíková, K., and Nakatani, T. (2017). Deep clustering-based beamforming for separation with unknown number of sources. In *Proc. Interspeech*, pages 1183–1187.

[Himawan et al., 2011] Himawan, I., McCowan, I., and Sridharan, S. (2011). Clustered blind beamforming from ad-hoc microphone arrays. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(4):661–676.

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

[Horiguchi et al., 2022a] Horiguchi, S., Fujita, Y., Watanabe, S., Xue, Y., and García, P. (2022a). Encoder-decoder based attractors for end-to-end neural diarization. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:1493–1507.

[Horiguchi et al., 2022b] Horiguchi, S., Takashima, Y., García, P., Watanabe, S., and Kawaguchi, Y. (2022b). Multi-channel end-to-end neural diarization with distributed microphones. In *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7332–7336.

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

[Hu et al., 2020a] Hu, J., Shen, L., Albanie, S., Sun, G., and Wu, E. (2020a). Squeeze-and-excitation networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(8):2011–2023.

[Hu et al., 2020b] Hu, Y., Liu, Y., Lv, S., Xing, M., Zhang, S., Fu, Y., Wu, J., Zhang, B., and Xie, L. (2020b). DCCRN: Deep complex convolution recurrent network for phase-aware speech enhancement. In *Proc. Interspeech*, pages 2472–2476.

[Huang et al., 2014] Huang, P.-S., Kim, M., Hasegawa-Johnson, M., and Smaragdis, P. (2014). Deep learning for monaural speech separation. In *ICASSP 2014 - 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1562–1566.

[Huh et al., 2023] Huh, J., Brown, A., weon Jung, J., Chung, J. S., Nagrani, A., Garcia-Romero, D., and Zisserman, A. (2023). VoxSRC 2022: The fourth VoxCeleb speaker recognition challenge. *arXiv preprint*. arXiv:2302.10248 [cs.SD].

[Huh et al., 2024] Huh, J., Chung, J. S., Nagrani, A., Brown, A., Jung, J.-w., Garcia-Romero, D., and Zisserman, A. (2024). The VoxCeleb speaker recognition challenge: A retrospective. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 32:3850–3866.

[Hummersone et al., 2014] Hummersone, C., Stokes, T., and Brookes, T. (2014). On the ideal ratio mask as the goal of computational auditory scene analysis. In Naik, G. R. and Wang, W., editors, *Blind Source Separation: Advances in Theory, Algorithms and Applications*, pages 349–368. Springer Berlin Heidelberg, Berlin, Heidelberg.

[International Telecommunication Union, 1996] International Telecommunication Union (1996). Recommendation ITU-T P.800: Methods for Subjective Determination of Transmission Quality. ITU-T Recommendation P.800, International Telecommunication Union.

[International Telecommunication Union, 2001] International Telecommunication Union (2001). Recommendation ITU-T P.862: Perceptual evaluation of speech quality (PESQ): An objective method for end-to-end speech quality assessment of narrow-band telephone networks and speech codecs. ITU-T Recommendation P.862, International Telecommunication Union.

[International Telecommunication Union, 2003] International Telecommunication Union (2003). Recommendation ITU-T P.835: Subjective Test Methodology for Evaluating Speech Communication Systems that Include Noise Suppression Algorithm. ITU-T Recommendation P.835, International Telecommunication Union.

[International Telecommunication Union, 2007] International Telecommunication Union (2007). Recommendation ITU-T P.862.2: Wideband extension to Recommendation P.862 for the assessment of wideband telephone networks and speech codecs. ITU-T Recommendation P.862.2, International Telecommunication Union.

[International Telecommunication Union, 2018] International Telecommunication Union (2018). Recommendation ITU-T P.863: Perceptual objective listening quality prediction. ITU-T Recommendation P.863, International Telecommunication Union.

[International Telecommunication Union, 2021] International Telecommunication Union (2021). Recommendation ITU-T P.808: Subjective Evaluation of Speech Quality with a Crowdsourcing Approach. ITU-T Recommendation P.808, International Telecommunication Union.

[Jan et al., 1995] Jan, E., Svaizer, P., and Flanagan, J. (1995). Matched-filter processing of microphone array for spatial volume selectivity. In *Proceedings of ISCAS '95 - International Symposium on Circuits and Systems*, volume 2, pages 1460–1463.

[Jensen and Taal, 2016] Jensen, J. and Taal, C. H. (2016). An algorithm for predicting the intelligibility of speech masked by modulated noise maskers. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(11):2009–2022.

[Jeub et al., 2009] Jeub, M., Schafer, M., and Vary, P. (2009). A binaural room impulse response database for the evaluation of dereverberation algorithms. In *2009 16th International Conference on Digital Signal Processing*, pages 1–5.

[Kenny, 2010] Kenny, P. (2010). Bayesian speaker verification with heavy-tailed priors. In *The Speaker and Language Recognition Workshop (Odyssey 2010)*.

[Kenny et al., 2005] Kenny, P., Boulianne, G., and Dumouchel, P. (2005). Eigenvoice modeling with sparse training data. *IEEE Transactions on Speech and Audio Processing*, 13(3):345–354.

[Kenny et al., 2007] Kenny, P., Boulianne, G., Ouellet, P., and Dumouchel, P. (2007). Joint factor analysis versus eigenchannels in speaker recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(4):1435–1447.

[Knapp and Carter, 1976] Knapp, C. and Carter, G. (1976). The generalized correlation method for estimation of time delay. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 24(4):320–327.

[Kolbæk et al., 2017] Kolbæk, M., Yu, D., Tan, Z.-H., and Jensen, J. (2017). Multitalker speech separation with utterance-level permutation invariant training of deep recurrent neural networks. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 25(10):1901–1913.

[Kumar et al., 2023] Kumar, A., Tan, K., Ni, Z., Manocha, P., Zhang, X., Henderson, E., and Xu, B. (2023). Torchaudio-Squim: Reference-less speech quality and intelligibility measures in torchaudio. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5.

[Kuttruff, 2009] Kuttruff, H. (2009). *Room acoustics*. CRC Press, 5th ed. edition. ISBN: 978-0415480215.

[Lea et al., 2017] Lea, C., Flynn, M. D., Vidal, R., Reiter, A., and Hager, G. D. (2017). Temporal convolutional networks for action segmentation and detection. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1003–1012.

[Lee et al., 2013] Lee, K. A., Larcher, A., You, C. H., Ma, B., and Li, H. (2013). Multi-session PLDA scoring of i-vector for partially open-set speaker detection. In *Proc. Interspeech*, pages 3651–3655.

[Leglaive et al., 2025] Leglaive, S., Fraticelli, M., ElGhazaly, H., Borne, L., Sadeghi, M., Wisdom, S., Pariente, M., Hershey, J. R., Pressnitzer, D., and Barker, J. P. (2025). Objective and subjective evaluation of speech enhancement methods in the UDASE task of the 7th CHiME challenge. *Computer Speech & Language*, 89.

[Li et al., 2016] Li, B., Sainath, T. N., Weiss, R. J., Wilson, K. W., and Bacchiani, M. (2016). Neural network adaptive beamforming for robust multichannel speech recognition. In *Proc. Interspeech*, pages 1976–1980.

[Liang et al., 2021] Liang, C., Chen, J., Guan, S., and Zhang, X.-L. (2021). Attention-based multi-channel speaker verification with ad-hoc microphone arrays. In *Proc. APSIPA Annual Summit and Conference*, pages 1111–1115.

[Liang et al., 2022] Liang, C., Chen, Y., Yao, J., and Zhang, X.-L. (2022). Multi-channel far-field speaker verification with large-scale ad-hoc microphone arrays. In *Proc. Interspeech*, pages 3679–3683.

[Lin et al., 1994] Lin, Q., Jan, E.-E., and Flanagan, J. (1994). Microphone arrays and speaker identification. *IEEE Transactions on Speech and Audio Processing*, 2(4):622–629.

[Lin et al., 2024a] Lin, Y., Cheng, M., Zhang, F., Gao, Y., Zhang, S., and Li, M. (2024a). VoxBlink2: A 100k+ speaker recognition corpus and the open-set speaker-identification benchmark. In *Proc. Interspeech*, pages 4263–4267.

[Lin et al., 2024b] Lin, Y., Qin, X., Zhao, G., Cheng, M., Jiang, N., Wu, H., and Li, M. (2024b). VoxBlink: A large scale speaker verification dataset on camera. In *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 10271–10275.

[Louizos et al., 2018] Louizos, C., Welling, M., and Kingma, D. P. (2018). Learning sparse neural networks through $L_0$ regularization. In *International Conference on Learning Representations*.

[Lozano-Diez et al., 2016] Lozano-Diez, A., Silnova, A., Matějka, P., Glembek, O., Plchot, O., Pešán, J., Burget, L., and Gonzalez-Rodriguez, J. (2016). Analysis and optimization of bottleneck features for speaker recognition. In *The Speaker and Language Recognition Workshop (Odyssey 2016)*, pages 352–357.

[Lu et al., 2013] Lu, X., Tsao, Y., Matsuda, S., and Hori, C. (2013). Speech enhancement based on deep denoising autoencoder. In *Proc. Interspeech*, pages 436–440.

[Luo et al., 2020] Luo, Y., Chen, Z., Mesgarani, N., and Yoshioka, T. (2020). End-to-end microphone permutation and number invariant multi-channel speech separation. In

*ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6394–6398.

[Luo et al., 2020] Luo, Y., Chen, Z., and Yoshioka, T. (2020). Dual-Path RNN: Efficient long sequence modeling for time-domain single-channel speech separation. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 46–50.

[Luo and Gu, 2024] Luo, Y. and Gu, R. (2024). Fast random approximation of multi-channel room impulse response. In *2024 IEEE International Conference on Acoustics, Speech, and Signal Processing Workshops (ICASSPW)*, pages 449–454.

[Luo et al., 2019] Luo, Y., Han, C., Mesgarani, N., Ceolini, E., and Liu, S.-C. (2019). FaSNet: Low-latency adaptive beamforming for multi-microphone audio processing. In *2019 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 260–267.

[Luo and Mesgarani, 2018a] Luo, Y. and Mesgarani, N. (2018a). Real-time single-channel dereverberation and separation with time-domain audio separation network. In *Proc. Interspeech*, pages 342–346.

[Luo and Mesgarani, 2018b] Luo, Y. and Mesgarani, N. (2018b). TaSNet: Time-domain audio separation network for real-time, single-channel speech separation. In *ICASSP 2018 - 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 696–700.

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

[Luong et al., 2015] Luong, T., Pham, H., and Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. In Màrquez, L., Callison-Burch, C., and Su, J., editors, *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

[Martin et al., 1997] Martin, A., Doddington, G., Kamm, T., Ordowski, M., and Przybocki, M. (1997). The DET curve in assessment of detection task performance. In *Proc. Eurospeech*, pages 1895–1898.

[Martin and Greenberg, 2010] Martin, A. F. and Greenberg, C. S. (2010). The NIST 2010 speaker recognition evaluation. In *Proc. Interspeech*, pages 2726–2729.

[Masuyama et al., 2023a] Masuyama, Y., Chang, X., Cornell, S., Watanabe, S., and Ono, N. (2023a). End-to-end integration of speech recognition, dereverberation, beamforming, and self-supervised learning representation. In *2022 IEEE Spoken Language Technology Workshop (SLT)*, pages 260–265.

[Masuyama et al., 2023b] Masuyama, Y., Chang, X., Zhang, W., Cornell, S., Wang, Z.-Q., Ono, N., Qian, Y., and Watanabe, S. (2023b). Exploring the integration of speech separation and recognition with self-supervised learning representation. In *2023 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 260–265.

[Matějka et al., 2019] Matějka, P., Plchot, O., Zeinali, H., Mošner, L., Silnova, A., Burget, L., and Glembek, O. (2019). Analysis of BUT submission in far-field scenarios of VOiCES 2019 challenge. In *Proc. Interspeech*, pages 2448–2452.

[McCowan, 2001] McCowan, I. (2001). Microphone arrays: A tutorial. *Queensland University, Australia*, pages 1–38.

[McCowan et al., 2000] McCowan, I., Moore, D., and Sridharan, S. (2000). Speech enhancement using near-field superdirectivity with an adaptive sidelobe canceler and post-filter. In *Proceedings of the 2000 Australian International Conference on Speech Science and Technology*, pages 268–273.

[McCowan et al., 2001] McCowan, I. A., Pelecanos, J., and Sridharan, S. (2001). Robust speaker recognition using microphone arrays. In *The Speaker and Language Recognition Workshop (Odyssey 2001)*, pages 101–106.

[McDonough and Kumatani, 2012] McDonough, J. and Kumatani, K. (2012). *Microphone Arrays*, chapter 6, pages 109–157. John Wiley & Sons.

[Mošner et al., 2023] Mošner, L., Plchot, O., Peng, J., Burget, L., and Černocký, J. (2023). Multi-channel speech separation with cross-attention and beamforming. In *Proc. Interspeech*, pages 1693–1697.

[Mošner et al., 2018a] Mošner, L., Matějka, P., Novotný, O., and Černocký, J. (2018a). Dereverberation and beamforming in far-field speaker recognition. In *ICASSP 2018 - 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5254–5258.

[Mošner et al., 2025] Mošner, L., Plchot, O., Burget, L., Zhang, C., Černocký, J., and Yu, M. (2025). Trainable multi-channel front-ends for joint beamforming and speaker embedding extraction. *Under review*.

[Mošner et al., 2022a] Mošner, L., Plchot, O., Burget, L., and Černocký, J. (2022a). Multi-channel speaker verification with Conv-TasNet based beamformer. In *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7982–7986.

[Mošner et al., 2022b] Mošner, L., Plchot, O., Burget, L., and Černocký, J. (2022b). MultiSV: Dataset for far-field multi-channel speaker verification. In *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7977–7981.

[Mošner et al., 2018b] Mošner, L., Plchot, O., Matějka, P., Novotný, O., and Černocký, J. (2018b). Dereverberation and beamforming in robust far-field speaker recognition. In *Proc. Interspeech*, pages 1334–1338.

[Mošner et al., 2019] Mošner, L., Plchot, O., Rohdin, J., Burget, L., and Černocký, J. (2019). Speaker verification with application-aware beamforming. In *2019 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 411–418.

[Mošner et al., 2020] Mošner, L., Plchot, O., Rohdin, J., and Černocký, J. (2020). Utilizing VOiCES dataset for multichannel speaker verification with beamforming. In *The Speaker and Language Recognition Workshop (Odyssey 2020)*, pages 187–193.

[Mošner et al., 2024] Mošner, L., Serizel, R., Burget, L., Plchot, O., Vincent, E., Peng, J., and Černocký, J. (2024). Multi-channel extension of pre-trained models for speaker verification. In *Proc. Interspeech*, pages 2135–2139.

[Nagrani et al., 2020a] Nagrani, A., Chung, J. S., Huh, J., Brown, A., Coto, E., Xie, W., McLaren, M., Reynolds, D. A., and Zisserman, A. (2020a). VoxSRC 2020: The second VoxCeleb speaker recognition challenge. *arXiv preprint*. arXiv:2012.06867 [cs.SD].

[Nagrani et al., 2020b] Nagrani, A., Chung, J. S., Xie, W., and Zisserman, A. (2020b). VoxCeleb: Large-scale speaker verification in the wild. *Computer Speech & Language*, 60:101027.

[Nagrani et al., 2017] Nagrani, A., Chung, J. S., and Zisserman, A. (2017). VoxCeleb: A large-scale speaker identification dataset. In *Proc. Interspeech*.

[Nair and Hinton, 2010] Nair, V. and Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, page 807–814, Madison, WI, USA. Omnipress.

[Nakatani et al., 2010] Nakatani, T., Yoshioka, T., Kinoshita, K., Miyoshi, M., and Juang, B.-H. (2010). Speech dereverberation based on variance-normalized delayed linear prediction. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(7):1717–1731.

[Nandwana et al., 2019] Nandwana, M. K., van Hout, J., McLaren, M., Richey, C., Lawson, A., and Barrios, M. A. (2019). The VOiCES from a distance challenge 2019 evaluation plan. *arXiv preprint*. arXiv:1902.10828 [eess.AS].

[Novoselov et al., 2023] Novoselov, S., Lavrentyeva, G., Avdeeva, A., Volokhov, V., Khmelev, N., Akulov, A., and Leonteva, P. (2023). On the robustness of wav2vec 2.0 based speaker recognition systems. In *Proc. Interspeech*, pages 3177–3181.

[Ochiai et al., 2020] Ochiai, T., Delcroix, M., Ikeshita, R., Kinoshita, K., Nakatani, T., and Araki, S. (2020). Beam-TasNet: Time-domain audio separation network meets frequency-domain beamformer. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6384–6388.

[Panayotov et al., 2015] Panayotov, V., Chen, G., Povey, D., and Khudanpur, S. (2015). LibriSpeech: An ASR corpus based on public domain audio books. In *ICASSP 2015 - 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210.

[Pandey and Wang, 2021] Pandey, A. and Wang, D. (2021). Dense CNN with self-attention for time-domain speech enhancement. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:1270–1279.

[Pascual et al., 2017] Pascual, S., Bonafonte, A., and Serrà, J. (2017). SEGAN: Speech enhancement generative adversarial network. In *Proc. Interspeech*, pages 3642–3646.

[Peddinti et al., 2015] Peddinti, V., Povey, D., and Khudanpur, S. (2015). A time delay neural network architecture for efficient modeling of long temporal contexts. In *Proc. Interspeech*, pages 3214–3218.

[Peer et al., 2008] Peer, I., Rafaely, B., and Zigel, Y. (2008). Room acoustics parameters affecting speaker recognition degradation under reverberation. In *2008 Hands-Free Speech Communication and Microphone Arrays*, pages 136–139.

[Peng et al., 2023a] Peng, J., Plchot, O., Stafylakis, T., Mošner, L., Burget, L., and Černocký, J. (2023a). An attention-based backend allowing efficient fine-tuning of transformer models for speaker verification. In *2022 IEEE Spoken Language Technology Workshop (SLT)*, pages 555–562.

[Peng et al., 2023b] Peng, J., Plchot, O., Stafylakis, T., Mošner, L., Burget, L., and Černocký, J. (2023b). Improving speaker verification with self-pretrained transformer models. In *Proc. Interspeech*, pages 5361–5365.

[Plchot et al., 2016] Plchot, O., Burget, L., Aronowitz, H., and Matějka, P. (2016). Audio enhancing with DNN autoencoder for speaker recognition. In *ICASSP 2016 - 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5090–5094.

[Povey et al., 2018] Povey, D., Cheng, G., Wang, Y., Li, K., Xu, H., Yarmohammadi, M., and Khudanpur, S. (2018). Semi-orthogonal low-rank matrix factorization for deep neural networks. In *Proc. Interspeech*, pages 3743–3747.

[Povey et al., 2011] Povey, D., Ghoshal, A., Boulianne, G., Burget, L., Glembek, O., Goel, N., Hannemann, M., Motlíček, P., Qian, Y., Schwarz, P., et al. (2011). The Kaldi speech recognition toolkit. In *2011 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 1–4. IEEE Signal Processing Society.

[Prince and Elder, 2007] Prince, S. J. and Elder, J. H. (2007). Probabilistic linear discriminant analysis for inferences about identity. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8.

[Przybocki and Martin, 2004] Przybocki, M. and Martin, A. F. (2004). NIST speaker recognition evaluation chronicles. In *The Speaker and Language Recognition Workshop (Odyssey 2004)*, pages 15–22.

[Qin et al., 2020] Qin, X., Bu, H., and Li, M. (2020). HI-MIA: A far-field text-dependent speaker verification database and the baselines. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7609–7613.

[Qin et al., 2023] Qin, X., Cai, D., and Li, M. (2023). Robust multi-channel far-field speaker verification under different in-domain data availability scenarios. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31:71–85.

[Qin et al., 2020] Qin, X., Li, M., Bu, H., Rao, W., Das, R. K., Narayanan, S., and Li, H. (2020). The INTERSPEECH 2020 Far-field speaker verification challenge. In *Proc. Interspeech*, pages 3456–3460.

[Reddy et al., 2021] Reddy, C. K. A., Gopal, V., and Cutler, R. (2021). DNSMOS: A non-intrusive perceptual objective speech quality metric to evaluate noise suppressors. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6493–6497.

[Reddy et al., 2022] Reddy, C. K. A., Gopal, V., and Cutler, R. (2022). DNSMOS P.835: A non-intrusive perceptual objective speech quality metric to evaluate noise suppressors. In *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 886–890.

[Reynolds and Rose, 1995] Reynolds, D. and Rose, R. (1995). Robust text-independent speaker identification using Gaussian mixture speaker models. *IEEE Transactions on Speech and Audio Processing*, 3(1):72–83.

[Richey et al., 2018] Richey, C., Barrios, M. A., Armstrong, Z., Bartels, C., Franco, H., Graciarena, M., Lawson, A., Nandwana, M. K., Stauffer, A., and van Hout, J. (2018). Voices obscured in complex environmental settings (VOICES) corpus. *arXiv preprint*. arXiv:1804.05053 [cs.SD].

[Rix et al., 2001] Rix, A., Beerends, J., Hollier, M., and Hekstra, A. (2001). Perceptual evaluation of speech quality (PESQ) - A new method for speech quality assessment of telephone networks and codecs. In *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.01CH37221)*, volume 2, pages 749–752.

[Rohdin et al., 2018] Rohdin, J., Silnova, A., Diez, M., Plchot, O., Matějka, P., and Burget, L. (2018). End-to-end DNN based speaker recognition inspired by i-vector and PLDA. In *ICASSP 2018 - 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4874–4878.

[Rohdin et al., 2020] Rohdin, J., Silnova, A., Diez, M., Plchot, O., Matějka, P., Burget, L., and Glembek, O. (2020). End-to-end DNN based text-independent speaker recognition for long and short utterances. *Computer Speech & Language*, 59:22–35.

[Roux et al., 2019] Roux, J. L., Wisdom, S., Erdogan, H., and Hershey, J. R. (2019). SDR – Half-baked or well done? In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 626–630.

[Sadjadi et al., 2020] Sadjadi, S. O., Greenberg, C., Singer, E., Olson, D., Mason, L., and Hernandez-Cordero, J. (2020). The 2019 NIST audio-visual speaker recognition evaluation. In *The Speaker and Language Recognition Workshop (Odyssey 2020)*. The Speaker and Language Recognition Workshop (Odyssey 2020), Tokyo.

[Safari et al., 2020] Safari, P., India, M., and Hernando, J. (2020). Self-attention encoding and pooling for speaker recognition. In *Proc. Interspeech*, pages 941–945.

[Sainath et al., 2017] Sainath, T. N., Weiss, R. J., Wilson, K. W., Li, B., Narayanan, A., Variani, E., Bacchiani, M., Shafran, I., Senior, A., Chin, K., Misra, A., and Kim, C. (2017). Multichannel signal processing with deep neural networks for automatic speech recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 25(5):965–979.

[Sainath et al., 2016] Sainath, T. N., Weiss, R. J., Wilson, K. W., Narayanan, A., and Bacchiani, M. (2016). Factored spatial and spectral multichannel raw waveform CLDNNs. In *ICASSP 2016 - 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

[Serizel et al., 2014] Serizel, R., Moonen, M., Van Dijk, B., and Wouters, J. (2014). Low-rank approximation based multichannel Wiener filter algorithms for noise reduction with application in cochlear implants. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 22(4):785–799.

[Shaw et al., 2018] Shaw, P., Uszkoreit, J., and Vaswani, A. (2018). Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 464–468, New Orleans, Louisiana. Association for Computational Linguistics.

[Shon et al., 2019] Shon, S., Tang, H., and Glass, J. (2019). VoiceID loss: Speech enhancement for speaker verification. In *Proc. Interspeech*, pages 2888–2892.

[Sifre, 2014] Sifre, L. (2014). *Rigid-Motion Scattering for Image Classification*. Ph.D. thesis, École Polytechnique, Palaiseau, France.

[Silnova et al., 2023] Silnova, A., Brümmer, N., Swart, A., and Burget, L. (2023). Toroidal probabilistic spherical discriminant analysis. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5.

[Sizov et al., 2014] Sizov, A., Lee, K. A., and Kinnunen, T. (2014). Unifying probabilistic linear discriminant analysis variants in biometric authentication. In Fränti, P., Brown, G., Loog, M., Escolano, F., and Pelillo, M., editors, *Structural, Syntactic, and Statistical Pattern Recognition*, pages 464–475, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Snyder et al., 2015] Snyder, D., Chen, G., and Povey, D. (2015). MUSAN: A music, speech, and noise corpus. *arXiv preprint*. arXiv:1510.08484 [cs.SD].

[Snyder et al., 2017] Snyder, D., Garcia-Romero, D., Povey, D., and Khudanpur, S. (2017). Deep neural network embeddings for text-independent speaker verification. In *Proc. Interspeech*, pages 999–1003.

[Snyder et al., 2019] Snyder, D., Garcia-Romero, D., Sell, G., McCree, A., Povey, D., and Khudanpur, S. (2019). Speaker recognition for multi-speaker conversations using x-vectors. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5796–5800.

[Snyder et al., 2018] Snyder, D., Garcia-Romero, D., Sell, G., Povey, D., and Khudanpur, S. (2018). X-vectors: Robust DNN embeddings for speaker recognition. In *ICASSP 2018 - 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5329–5333.

[Snyder et al., 2016] Snyder, D., Ghahremani, P., Povey, D., Garcia-Romero, D., Carmiel, Y., and Khudanpur, S. (2016). Deep neural network-based speaker embeddings for end-to-end speaker verification. In *2016 IEEE Spoken Language Technology Workshop (SLT)*, pages 165–170.

[Soong et al., 1985] Soong, F., Rosenberg, A., Rabiner, L., and Juang, B. (1985). A vector quantization approach to speaker recognition. In *ICASSP 1985 - 1985 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 10, pages 387–390.

[Souden et al., 2010] Souden, M., Benesty, J., and Affes, S. (2010). On optimal frequency-domain multichannel linear filtering for noise reduction. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(2):260–276.

[Stafylakis et al., 2023] Stafylakis, T., Mošner, L., Kakouros, S., Plchot, O., Burget, L., and Černocký, J. (2023). Extracting speaker and emotion information from self-supervised speech models via channel-wise correlations. In *2022 IEEE Spoken Language Technology Workshop (SLT)*, pages 1136–1143.

[Stafylakis et al., 2021] Stafylakis, T., Rohdin, J., and Burget, L. (2021). Speaker embeddings by modeling channel-wise correlations. In *Proc. Interspeech*, pages 501–505.

[Subakan et al., 2021] Subakan, C., Ravanelli, M., Cornell, S., Bronzi, M., and Zhong, J. (2021). Attention is all you need in speech separation. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 21–25.

[Szőke et al., 2019] Szőke, I., Skácel, M., Mošner, L., Paliesek, J., and Černocký, J. (2019). Building and evaluation of a real room impulse response dataset. *IEEE Journal of Selected Topics in Signal Processing*, 13(4):863–876.

[Taal et al., 2010] Taal, C. H., Hendriks, R. C., Heusdens, R., and Jensen, J. (2010). A short-time objective intelligibility measure for time-frequency weighted noisy speech. In *ICASSP 2010 - 2010 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4214–4217.

[Taal et al., 2011] Taal, C. H., Hendriks, R. C., Heusdens, R., and Jensen, J. (2011). An algorithm for intelligibility prediction of time–frequency weighted noisy speech. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(7):2125–2136.

[Taherian et al., 2020] Taherian, H., Wang, Z.-Q., Chang, J., and Wang, D. (2020). Robust speaker recognition based on single-channel and multi-channel speech enhancement. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:1293–1302.

[Taherian et al., 2019] Taherian, H., Wang, Z.-Q., and Wang, D. (2019). Deep learning based multi-channel speaker recognition in noisy and reverberant environments. In *Proc. Interspeech*, pages 4070–4074.

[Tan and Wang, 2019] Tan, K. and Wang, D. (2019). Complex spectral mapping with a convolutional recurrent network for monaural speech enhancement. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6865–6869.

[Thienpondt et al., 2021] Thienpondt, J., Desplanques, B., and Demuynck, K. (2021). The IdLab VoxSRC-20 submission: Large margin fine-tuning and quality-aware score calibration in DNN based speaker verification. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5814–5818.

[Trees, 2002] Trees, H. L. V. (2002). *Optimum Array Processing.* John Wiley & Sons, New York, USA. ISBN: 0471093904.

[Variani et al., 2014] Variani, E., Lei, X., McDermott, E., Moreno, I. L., and Gonzalez-Dominguez, J. (2014). Deep neural networks for small footprint text-dependent speaker verification. In *ICASSP 2014 - 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4052–4056.

[Vaswani et al., 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. (2017). Attention is all you need. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

[Villalba et al., 2020] Villalba, J., Chen, N., Snyder, D., Garcia-Romero, D., McCree, A., Sell, G., Borgstrom, J., García-Perera, L. P., Richardson, F., Dehak, R., Torres-Carrasquillo, P. A., and Dehak, N. (2020). State-of-the-art speaker recognition with neural network embeddings in NIST SRE18 and Speakers in the wild evaluations. *Computer Speech & Language*, 60.

[Vincent et al., 2006] Vincent, E., Gribonval, R., and Fevotte, C. (2006). Performance measurement in blind audio source separation. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(4):1462–1469.

[Waibel et al., 1989] Waibel, A., Hanazawa, T., Hinton, G., Shikano, K., and Lang, K. (1989). Phoneme recognition using time-delay neural networks. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37(3):328–339.

[Wang, 2005] Wang, D. (2005). On ideal binary mask as the computational goal of auditory scene analysis. In Divenyi, P., editor, *Speech Separation by Humans and Machines*, pages 181–197. Springer US, Boston, MA.

[Wang and Brown, 2006] Wang, D. and Brown, G. J. (2006). *Computational Auditory Scene Analysis*. Wiley Interscience, Hoboken, N.J. ISBN: 9780471741091.

[Wang and Chen, 2018] Wang, D. and Chen, J. (2018). Supervised speech separation based on deep learning: An overview. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(10):1702–1726.

[Wang et al., 2020] Wang, D., Chen, Z., and Yoshioka, T. (2020). Neural speech separation using spatially distributed microphones. In *Proc. Interspeech*, pages 339–343.

[Wang et al., 2021a] Wang, D., Yoshioka, T., Chen, Z., Wang, X., Zhou, T., and Meng, Z. (2021a). Continuous speech separation with ad hoc microphone arrays. In *29th European Signal Processing Conference (EUSIPCO)*, pages 1100–1104.

[Wang et al., 2018] Wang, F., Cheng, J., Liu, W., and Liu, H. (2018). Additive margin softmax for face verification. *IEEE Signal Processing Letters*, 25(7):926–930.

[Wang et al., 2021b] Wang, S., Yang, Y., Qian, Y., and Yu, K. (2021b). Revisiting the statistics pooling layer in deep speaker embedding learning. In *2021 12th International Symposium on Chinese Spoken Language Processing (ISCSLP)*, pages 1–5.

[Wang et al., 2014] Wang, Y., Narayanan, A., and Wang, D. (2014). On training targets for supervised speech separation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 22(12):1849–1858.

[Wang and Wang, 2012a] Wang, Y. and Wang, D. (2012a). Boosting classification based speech separation using temporal dynamics. In *Proc. Interspeech*, pages 1528–1531.

[Wang and Wang, 2012b] Wang, Y. and Wang, D. (2012b). Cocktail party processing via structured prediction. In Pereira, F., Burges, C., Bottou, L., and Weinberger, K., editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc.

[Wang and Wang, 2013] Wang, Y. and Wang, D. (2013). Towards scaling up classification-based speech separation. *IEEE Transactions on Audio, Speech, and Language Processing*, 21(7):1381–1390.

[Wang et al., 2023a] Wang, Z.-Q., Cornell, S., Choi, S., Lee, Y., Kim, B.-Y., and Watanabe, S. (2023a). TF-GridNet: Integrating full- and sub-band modeling for speech separation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31:3221–3236.

[Wang et al., 2023b] Wang, Z.-Q., Cornell, S., Choi, S., Lee, Y., Kim, B.-Y., and Watanabe, S. (2023b). TF-GridNet: Making time-frequency domain models great again for monaural speaker separation. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5.

[Wang et al., 2021c] Wang, Z.-Q., Wang, P., and Wang, D. (2021c). Multi-microphone complex spectral mapping for utterance-wise and continuous speech separation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:2001–2014.

[Warsitz and Haeb-Umbach, 2007] Warsitz, E. and Haeb-Umbach, R. (2007). Blind acoustic beamforming based on generalized eigenvalue decomposition. *IEEE Transactions on Audio, Speech and Language Processing*, 15(5):1529–1539.

[Warsitz et al., 2008] Warsitz, E., Krueger, A., and Haeb-Umbach, R. (2008). Speech enhancement with a new generalized eigenvector blocking matrix for application in a generalized sidelobe canceller. In *ICASSP 2008 - 2008 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 73–76.

[Weninger et al., 2014] Weninger, F., Hershey, J. R., Le Roux, J., and Schuller, B. (2014). Discriminatively trained recurrent neural networks for single-channel speech separation. In *2014 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pages 577–581.

[Wichern et al., 2019] Wichern, G., Antognini, J., Flynn, M., Zhu, L. R., McQuinn, E., Crow, D., Manilow, E., and Roux, J. L. (2019). WHAM!: Extending speech separation to noisy environments. In *Proc. Interspeech*, pages 1368–1372.

[Williamson et al., 2016] Williamson, D. S., Wang, Y., and Wang, D. (2016). Complex ratio masking for monaural speech separation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(3):483–492.

[Woelfel and McDonough, 2009] Woelfel, M. and McDonough, J. (2009). *Distant Speech Recognition*. John Wiley & Sons. ISBN: 9780470517048.

[Xiao et al., 2016] Xiao, X., Watanabe, S., Erdogan, H., Lu, L., Hershey, J., Seltzer, M. L., Chen, G., Zhang, Y., Mandel, M., and Yu, D. (2016). Deep beamforming networks for multi-channel speech recognition. In *ICASSP 2016 - 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5745–5749.

[Xie et al., 2017] Xie, S., Girshick, R., Dollár, P., Tu, Z., and He, K. (2017). Aggregated residual transformations for deep neural networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5987–5995.

[Xie et al., 2019] Xie, W., Nagrani, A., Chung, J. S., and Zisserman, A. (2019). Utterance-level aggregation for speaker recognition in the wild. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5791–5795.

[Xu et al., 2014] Xu, Y., Du, J., Dai, L.-R., and Lee, C.-H. (2014). An experimental study on speech enhancement based on deep neural networks. *IEEE Signal Processing Letters*, 21(1):65–68.

[Xu et al., 2019] Xu, Y., Weng, C., Hui, L., Liu, J., Yu, M., Su, D., and Yu, D. (2019). Joint training of complex ratio mask based beamformer and acoustic model for noise robust ASR. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6745–6749.

[Xu et al., 2021] Xu, Y., Zhang, Z., Yu, M., Zhang, S.-X., and Yu, D. (2021). Generalized spatio-temporal RNN beamformer for target speech separation. In *Proc. Interspeech*, pages 3076–3080.

[Yang and Chang, 2019] Yang, J.-Y. and Chang, J.-H. (2019). Joint optimization of neural acoustic beamforming and dereverberation with x-vectors for robust speaker verification. In *Proc. Interspeech*, pages 4075–4079.

[Yang et al., 2021] Yang, S.-w., Chi, P.-H., Chuang, Y.-S., Lai, C.-I. J., Lakhotia, K., Lin, Y. Y., Liu, A. T., Shi, J., Chang, X., Lin, G.-T., Huang, T.-H., Tseng, W.-C., Lee, K.-t., Liu, D.-R., Huang, Z., Dong, S., Li, S.-W., Watanabe, S., Mohamed, A., and Lee, H.-y. (2021). SUPERB: Speech processing universal performance benchmark. In *Proc. Interspeech*, pages 1194–1198.

[Yoshioka et al., 2015] Yoshioka, T., Ito, N., Delcroix, M., et al. (2015). The NTT CHiME-3 system: Advances in speech enhancement and recognition for mobile multi-microphone devices. In *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 436–443.

[Yoshioka and Nakatani, 2012] Yoshioka, T. and Nakatani, T. (2012). Generalization of multi-channel linear prediction methods for blind MIMO impulse response shortening. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(10):2707–2720.

[Yu et al., 2017] Yu, D., Kolbæk, M., Tan, Z.-H., and Jensen, J. (2017). Permutation invariant training of deep models for speaker-independent multi-talker speech separation. In *ICASSP 2017 - 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 241–245.

[Zeinali et al., 2019] Zeinali, H., Wang, S., Silnova, A., Matějka, P., and Plchot, O. (2019). BUT system description to VoxCeleb speaker recognition challenge 2019. In *Proceedings of The VoxCeleb Challange Workshop 2019*, pages 1–4, Graz.

[Zhang et al., 2020] Zhang, J., Zorilă, C., Doddipatla, R., and Barker, J. (2020). On end-to-end multi-channel time domain speech separation in reverberant environments. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6389–6393.

[Zhang et al., 2024] Zhang, W., Scheibler, R., Saijo, K., Cornell, S., Li, C., Ni, Z., Pirklbauer, J., Sach, M., Watanabe, S., Fingscheidt, T., and Qian, Y. (2024). URGENT challenge: Universality, robustness, and generalizability for speech enhancement. In *Proc. Interspeech*, pages 4868–4872.

[Zhang et al., 2021a] Zhang, Z., Xu, Y., Yu, M., Zhang, S.-X., Chen, L., Williamson, D. S., and Yu, D. (2021a). Multi-channel multi-frame ADL-MVDR for target speech separation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3526–3540.

[Zhang et al., 2021b] Zhang, Z., Xu, Y., Yu, M., Zhang, S.-X., Chen, L., and Yu, D. (2021b). ADL-MVDR: All deep learning MVDR beamformer for target speech separation. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6089–6093.

[Zhong et al., 2019] Zhong, Y., Arandjelović, R., and Zisserman, A. (2019). GhostVLAD for set-based face recognition. In *Computer Vision – ACCV 2018*, pages 35–50. Springer International Publishing, Cham.

[Zhu et al., 2024] Zhu, Q., Zhang, J., Gu, Y., Hu, Y., and Dai, L. (2024). Multichannel AV-wav2vec2: A framework for learning multichannel multi-modal speech representation. In *Annual AAAI Conference on Artificial Intelligence*, pages 19768–19776.

[Zhu et al., 2018] Zhu, Y., Ko, T., Snyder, D., Mak, B., and Povey, D. (2018). Self-attentive speaker embeddings for text-independent speaker verification. In *Proc. Interspeech*, pages 3573–3577.

[Ziomek, 1994] Ziomek, L. (1994). *Fundamentals of Acoustic Field Theory and Space-Time Signal Processing*. CRC Press. ISBN: 0849394554.

# Appendix A

# I-vector Derivations

## A.1 Data Likelihood Under the GMM Model

Let $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\}$ be $T$ feature vectors of an utterance and $\theta = \{\mathbf{w}, \boldsymbol{\mu}, \boldsymbol{\Sigma}\}$ parameters of a GMM with $C$ components, where

$$\mathbf{w} = \begin{bmatrix} w_1 \\ \vdots \\ w_C \end{bmatrix}, \tag{A.1}$$

$$\boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_1 \\ \vdots \\ \boldsymbol{\mu}_C \end{bmatrix}, \tag{A.2}$$

$$\boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_1 & \mathbf{0} & \ldots & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_2 & \ldots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \ldots & \boldsymbol{\Sigma}_C \end{bmatrix}, \tag{A.3}$$

are a vector of component weights (i.e., priors), supervector of concatenated components' means, and a block diagonal matrix with diagonal blocks representing components' covariance matrices, respectively. The log-likelihood of the data under the GMM model (and the independence assumption) is expressed as

$$\log p(\mathcal{X}; \theta) = \sum_{t=1}^{T} \log p(\mathbf{x}_t; \theta) = \sum_{t=1}^{T} \log \sum_{c=1}^{C} w_c \mathcal{N}(\mathbf{x}_t; \boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c). \tag{A.4}$$

Following [Glembek, 2012], log-likelihood of the feature vector $\mathbf{x}_t$ (i.e., observed variable) generated by any model with hidden variable $c$[1] parametrized by $\theta$ can be in general expressed as

$$\log p(\mathbf{x}_t; \theta) = \mathrm{E}_{q(c)}[\log p(\mathbf{x}_t | c; \theta)] - \mathrm{D}_{\mathrm{KL}}(q(c) || p(c; \theta)) + \mathrm{D}_{\mathrm{KL}}(q(c) || p(c | \mathbf{x}_t; \theta)), \tag{A.5}$$

[1] In order to simplify notation, we use symbols $\mathbf{x}_t$ and $c$ (even though the formula is general).

where $q(c)$ is an arbitrary distribution of the latent variable. Rewriting (A.4) using (A.5), we obtain

$$\begin{aligned}\log p(\mathcal{X};\theta) = &\sum_{t=1}^{T}\sum_{c=1}^{C} q(c)\log\mathcal{N}(\mathbf{x}_t;\boldsymbol{\mu}_c,\boldsymbol{\Sigma}_c)\\ &-\sum_{t=1}^{T}\sum_{c=1}^{C} q(c)\log\frac{q(c)}{w_c}\\ &+\sum_{t=1}^{T}\sum_{c=1}^{C} q(c)\log\frac{q(c)}{\gamma_{c,t}}.\end{aligned} \tag{A.6}$$

We note that the symbol $\gamma_{c,t}$ (commonly referred to as *occupation probability*) was used for the posterior probability of the $c$th component given an observation $\mathbf{x}_t$:

$$\gamma_{c,t} := p(c|\mathbf{x}_t;\theta) = \frac{p(c;\theta)p(\mathbf{x}_t|c;\theta)}{p(\mathbf{x}_t;\theta)} = \frac{w_c\mathcal{N}(\mathbf{x}_t;\boldsymbol{\mu}_c,\boldsymbol{\Sigma}_c)}{\sum_{c'=1}^{C} w_{c'}\mathcal{N}(\mathbf{x}_t;\boldsymbol{\mu}_{c'},\boldsymbol{\Sigma}_{c'})}. \tag{A.7}$$

The configuration of the posterior probabilities is called *alignment* of the data to the GMM components. In the case when the $q(c)$ distribution matches the true posterior (i.e., $q(c) = \gamma_{c,t}$), the last Kullback-Leibler divergence term in (A.6) becomes 0, providing the likelihood of the form

$$\begin{aligned}\log p(\mathcal{X};\theta) = &\sum_{t=1}^{T}\sum_{c=1}^{C} \gamma_{c,t}\log\mathcal{N}(\mathbf{x}_t;\boldsymbol{\mu}_c,\boldsymbol{\Sigma}_c)\\ &-\sum_{t=1}^{T}\sum_{c=1}^{C} \gamma_{c,t}\log\frac{\gamma_{c,t}}{w_c}.\end{aligned} \tag{A.8}$$

This version of the likelihood is beneficial as the sum over components is no longer within the logarithm function (this will become useful immediately). By explicitly expressing the Gaussian probability density function, we obtain

$$\begin{aligned}\log p(\mathcal{X};\theta) = &\sum_{t=1}^{T}\sum_{c=1}^{C} \gamma_{c,t}\log\frac{1}{(2\pi)^{F/2}|\boldsymbol{\Sigma}_c|^{1/2}}\\ &+\sum_{t=1}^{T}\sum_{c=1}^{C} \gamma_{c,t}\left[-\frac{1}{2}(\mathbf{x}_t-\boldsymbol{\mu}_c)^{\mathsf{T}}\boldsymbol{\Sigma}_c^{-1}(\mathbf{x}_t-\boldsymbol{\mu}_c)\right]\\ &-\sum_{t=1}^{T}\sum_{c=1}^{C} \gamma_{c,t}\log\frac{\gamma_{c,t}}{w_c}.\end{aligned} \tag{A.9}$$

Let us now define sufficient statistics for completeness. They will become useful in the following text. Given an utterance $\mathcal{X}$ and computed occupation probabilities,

$$\begin{aligned}N_c &= \sum_{t=1}^{T}\gamma_{c,t},\\ \mathbf{f}_c &= \sum_{t=1}^{T}\gamma_{c,t}\mathbf{x}_t,\\ \mathbf{S}_c &= \sum_{t=1}^{T}\gamma_{c,t}\mathbf{x}_t\mathbf{x}_t^{\mathsf{T}}\end{aligned} \tag{A.10}$$

denote the zeroth-, first-, and second-order statistics. It is convenient to gather them to yield

$$\mathbf{N} = \begin{bmatrix} N_1\mathbf{I} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & N_2\mathbf{I} & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & N_C\mathbf{I} \end{bmatrix} \in \mathbb{R}^{CF\times CF},$$
$$\mathbf{f} = \begin{bmatrix} \mathbf{f}_1 \\ \vdots \\ \mathbf{f}_C \end{bmatrix} \in \mathbb{R}^{CF}, \tag{A.11}$$
$$\mathbf{S} = \begin{bmatrix} \mathbf{S}_1 & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{S}_2 & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{S}_C \end{bmatrix} \in \mathbb{R}^{CF\times CF}.$$

The benefit of the likelihood in the form of (A.9) now becomes clear as it allows us to identify sufficient statistics. Realizing that the terms $\sum_{t,c}\gamma_{c,t}\mathbf{x}_t^\intercal\boldsymbol{\Sigma}_c^{-1}\mathbf{x}_t$; $\sum_{t,c}\gamma_{c,t}\boldsymbol{\mu}_c^\intercal\boldsymbol{\Sigma}_c^{-1}\mathbf{x}_t$; and $\sum_{t,c}\gamma_{c,t}\boldsymbol{\mu}_c^\intercal\boldsymbol{\Sigma}_c^{-1}\boldsymbol{\mu}_c$ can be rewritten using (A.2), (A.3), and (A.11), we get

$$\begin{aligned}\log p(\mathcal{X};\theta) = &\sum_{c=1}^{C} N_c \log\frac{1}{(2\pi)^{F/2}|\boldsymbol{\Sigma}_c|^{1/2}} \\ &- \frac{1}{2}\mathrm{tr}(\boldsymbol{\Sigma}^{-1}\mathbf{S}) + \boldsymbol{\mu}^\intercal\boldsymbol{\Sigma}^{-1}\mathbf{f} - \frac{1}{2}\boldsymbol{\mu}^\intercal\mathbf{N}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} \\ &- \sum_{t=1}^{T}\sum_{c=1}^{C}\gamma_{c,t}\log\frac{\gamma_{c,t}}{w_c}.\end{aligned} \tag{A.12}$$

## A.2 Data Alignment Using Universal Background Model

(A.12) is the true log-likelihood of the data since the data alignment given by $\gamma_{c,t}$ corresponds to the correct component posterior. The form of the likelihood in (A.6) provided us with an insight that as long as $q(c)$ does not equal the true posterior, the KL-divergence term becomes non-zero. In the cases when the actual alignment is missing but a different model (that is believed to be a good approximation) to extract it exists, (A.12) is still employed, bearing in mind that the non-zero KL-divergence term has been omitted (because it cannot be computed without the true alignment). Then, the obtained log-likelihood serves as a lower bound to the true log-likelihood. In the context of i-vectors, the Universal Background Model (UBM) trained on speech of all available speakers is used to provide alignment and, in turn, sufficient statistics.

In the following text, we will assume that the alignment was obtained with the UBM, accepting that the log-likelihood approximates the true one. We will still use the same symbols to avoid cluttered notation. Recalling that the utterance-dependent mean supervector is modeled as $\boldsymbol{\mu} = \mathbf{u} + \mathbf{T}\boldsymbol{\varphi}$ in the i-vector approach, we can rewrite the lower-bound on

log-likelihood as

$$\begin{aligned}\log p(\mathcal{X}|\boldsymbol{\varphi};\theta,\mathbf{T}) = &\sum_{c=1}^{C} N_c \log \frac{1}{(2\pi)^{F/2}|\boldsymbol{\Sigma}_c|^{1/2}} \\ &- \frac{1}{2}\mathrm{tr}(\boldsymbol{\Sigma}^{-1}\mathbf{S}) + (\mathbf{u}+\mathbf{T}\boldsymbol{\varphi})^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\mathbf{f} \\ &- \frac{1}{2}[\mathbf{u}^{\mathsf{T}}\mathbf{N}\boldsymbol{\Sigma}^{-1}\mathbf{u} + \mathbf{u}^{\mathsf{T}}\mathbf{N}\boldsymbol{\Sigma}^{-1}\mathbf{T}\boldsymbol{\varphi} + \boldsymbol{\varphi}^{\mathsf{T}}\mathbf{T}^{\mathsf{T}}\mathbf{N}\boldsymbol{\Sigma}^{-1}\mathbf{u} + \boldsymbol{\varphi}^{\mathsf{T}}\mathbf{T}^{\mathsf{T}}\mathbf{N}\boldsymbol{\Sigma}^{-1}\mathbf{T}\boldsymbol{\varphi}] \\ &- \sum_{t=1}^{T}\sum_{c=1}^{C} \gamma_{c,t} \log \frac{\gamma_{c,t}}{w_c}.\end{aligned} \tag{A.13}$$

It is convenient to center the sufficient statistics. It effectively makes the UBM mean supervector $\mathbf{u}$ a zero vector. Centering provides sufficient statistics in the form of

$$\begin{aligned}\tilde{\mathbf{f}}_c &= \sum_{t=1}^{T} \gamma_{c,t}(\mathbf{x}_t - \boldsymbol{\mu}_c) = \mathbf{f}_c - N_c\boldsymbol{\mu}_c, \\ \tilde{\mathbf{S}}_c &= \sum_{t=1}^{T} \gamma_{c,t}(\mathbf{x}_t - \boldsymbol{\mu}_c)(\mathbf{x}_t - \boldsymbol{\mu}_c)^{\mathsf{T}} = \mathbf{S}_c - \mathbf{f}_c\boldsymbol{\mu}_c^{\mathsf{T}} - \boldsymbol{\mu}_c\mathbf{f}_c^{\mathsf{T}} + N_c\boldsymbol{\mu}_c\boldsymbol{\mu}_c^{\mathsf{T}}.\end{aligned} \tag{A.14}$$

The gathered statistics $\tilde{\mathbf{f}}$ and $\tilde{\mathbf{S}}$ are defined analogically to those in (A.11). The centering provides the following simplification [Kenny et al., 2005]:

$$\begin{aligned}\log p(\mathcal{X}|\boldsymbol{\varphi};\theta,\mathbf{T}) = &\sum_{c=1}^{C} N_c \log \frac{1}{(2\pi)^{F/2}|\boldsymbol{\Sigma}_c|^{1/2}} \\ &- \frac{1}{2}\mathrm{tr}(\boldsymbol{\Sigma}^{-1}\tilde{\mathbf{S}}) + \boldsymbol{\varphi}^{\mathsf{T}}\mathbf{T}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\tilde{\mathbf{f}} - \frac{1}{2}\boldsymbol{\varphi}^{\mathsf{T}}\mathbf{T}^{\mathsf{T}}\mathbf{N}\boldsymbol{\Sigma}^{-1}\mathbf{T}\boldsymbol{\varphi} \\ &- \sum_{t=1}^{T}\sum_{c=1}^{C} \gamma_{c,t} \log \frac{\gamma_{c,t}}{w_c}.\end{aligned} \tag{A.15}$$

## A.3 I-Vector Posterior

Our aim is to obtain the posterior $p(\boldsymbol{\varphi}|\mathcal{X};\theta,\mathbf{T})$. Then the i-vector representing a sequence $\mathcal{X}$ of a variable length is a MAP point estimate of the posterior distribution, i.e., the mode (mean) of $p(\boldsymbol{\varphi}|\mathcal{X};\theta,\mathbf{T})$. Using Bayes' rule, we can express the posterior as

$$p(\boldsymbol{\varphi}|\mathcal{X};\theta,\mathbf{T}) \propto p(\mathcal{X}|\boldsymbol{\varphi};\theta,\mathbf{T})p(\boldsymbol{\varphi}). \tag{A.16}$$

Using (A.15) and the fact that $\boldsymbol{\varphi}$ has a standard normal prior according to (2.11), the logarithm of the joint probability becomes

$$\log[p(\mathcal{X}|\boldsymbol{\varphi};\theta,\mathbf{T})p(\boldsymbol{\varphi})] = \boldsymbol{\varphi}^{\mathsf{T}}\mathbf{T}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\tilde{\mathbf{f}} - \frac{1}{2}\boldsymbol{\varphi}^{\mathsf{T}}\mathbf{T}^{\mathsf{T}}\mathbf{N}\boldsymbol{\Sigma}^{-1}\mathbf{T}\boldsymbol{\varphi} - \frac{1}{2}\boldsymbol{\varphi}^{\mathsf{T}}\boldsymbol{\varphi} + \text{const}. \tag{A.17}$$

By completing squares, we realize that the i-vector posterior is a Gaussian distribution

$$p(\boldsymbol{\varphi}|\mathcal{X};\theta,\mathbf{T}) = \mathcal{N}(\boldsymbol{\varphi};\boldsymbol{\mu}_{\boldsymbol{\varphi}},\boldsymbol{\Gamma}_{\boldsymbol{\varphi}}^{-1}), \tag{A.18}$$

with the precision and mean defined as

$$\boldsymbol{\Gamma}_{\boldsymbol{\varphi}} = \mathbf{T}^{\intercal}\mathbf{N}\boldsymbol{\Sigma}^{-1}\mathbf{T} + \mathbf{I}, \tag{A.19}$$

$$\boldsymbol{\mu}_{\boldsymbol{\varphi}} = \boldsymbol{\Gamma}_{\boldsymbol{\varphi}}^{-1}\mathbf{T}^{\intercal}\boldsymbol{\Sigma}^{-1}\tilde{\mathbf{f}}. \tag{A.20}$$

Finally, the mean $\boldsymbol{\mu}_{\boldsymbol{\varphi}}$ represents the i-vector for the given utterance.

# Appendix B

# Derivations for Simplified PLDA

## B.1 Generative Equations

We will show that generative equations for simplified PLDA lead to the same expression of the LLR in terms of within-class and across-class covariance matrices as for the standard PLDA. Under the simplified PLDA model, an alternative to (2.34) is given as:

$$\begin{aligned} \mathcal{H}_0 : \begin{bmatrix} \boldsymbol{\varphi}_1 \\ \boldsymbol{\varphi}_2 \end{bmatrix} &= \begin{bmatrix} \mathbf{m} \\ \mathbf{m} \end{bmatrix} + \begin{bmatrix} \mathbf{U} \\ \mathbf{U} \end{bmatrix} \begin{bmatrix} \mathbf{y}_{12} \end{bmatrix} + \begin{bmatrix} \boldsymbol{\epsilon}_1 \\ \boldsymbol{\epsilon}_2 \end{bmatrix}, \\ \mathcal{H}_1 : \begin{bmatrix} \boldsymbol{\varphi}_1 \\ \boldsymbol{\varphi}_2 \end{bmatrix} &= \begin{bmatrix} \mathbf{m} \\ \mathbf{m} \end{bmatrix} + \begin{bmatrix} \mathbf{U} & \mathbf{0} \\ \mathbf{0} & \mathbf{U} \end{bmatrix} \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{bmatrix} + \begin{bmatrix} \boldsymbol{\epsilon}_1 \\ \boldsymbol{\epsilon}_2 \end{bmatrix}. \end{aligned} \tag{B.1}$$

Being equipped with the knowledge of how to determine the likelihood of $[\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2]^\intercal$ under the two hypotheses given the generative equations, we can directly write the expression for the LLR:

$$\begin{aligned} s(\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2) = &\log \mathcal{N}\left( \begin{bmatrix} \boldsymbol{\varphi}_1 \\ \boldsymbol{\varphi}_2 \end{bmatrix}; \begin{bmatrix} \mathbf{m} \\ \mathbf{m} \end{bmatrix}, \begin{bmatrix} \mathbf{U}\mathbf{U}^\intercal + \mathbf{E}^{-1} & \mathbf{U}\mathbf{U}^\intercal \\ \mathbf{U}\mathbf{U}^\intercal & \mathbf{U}\mathbf{U}^\intercal + \mathbf{E}^{-1} \end{bmatrix} \right) - \\ &\log \mathcal{N}\left( \begin{bmatrix} \boldsymbol{\varphi}_1 \\ \boldsymbol{\varphi}_2 \end{bmatrix}; \begin{bmatrix} \mathbf{m} \\ \mathbf{m} \end{bmatrix}, \begin{bmatrix} \mathbf{U}\mathbf{U}^\intercal + \mathbf{E}^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{U}\mathbf{U}^\intercal + \mathbf{E}^{-1} \end{bmatrix} \right). \end{aligned} \tag{B.2}$$

Recalling that the across-class covariance matrix is $\boldsymbol{\Sigma}_{ac} = \mathbf{U}\mathbf{U}^\intercal$ and within-class covariance matrix is $\boldsymbol{\Sigma}_{wc} = \mathbf{E}^{-1}$, we can rewrite the score formula as:

$$\begin{aligned} s(\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2) = &\log \mathcal{N}\left( \begin{bmatrix} \boldsymbol{\varphi}_1 \\ \boldsymbol{\varphi}_2 \end{bmatrix}; \begin{bmatrix} \mathbf{m} \\ \mathbf{m} \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma}_{tot} & \boldsymbol{\Sigma}_{ac} \\ \boldsymbol{\Sigma}_{ac} & \boldsymbol{\Sigma}_{tot} \end{bmatrix} \right) - \\ &\log \mathcal{N}\left( \begin{bmatrix} \boldsymbol{\varphi}_1 \\ \boldsymbol{\varphi}_2 \end{bmatrix}; \begin{bmatrix} \mathbf{m} \\ \mathbf{m} \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma}_{tot} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_{tot} \end{bmatrix} \right), \end{aligned} \tag{B.3}$$

which exactly corresponds to (2.38) derived in the context of the standard PLDA.

## B.2 Covariance Matrix Inversion

Score computation with a simplified PLDA requires inversion of the block covariance matrix in (B.2):

$$\boldsymbol{\Sigma}_0 = \begin{bmatrix} \boldsymbol{\Sigma}_{tot} & \boldsymbol{\Sigma}_{ac} \\ \boldsymbol{\Sigma}_{ac} & \boldsymbol{\Sigma}_{tot} \end{bmatrix} \qquad \boldsymbol{\Sigma}_0^{-1} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} \tag{B.4}$$

The inverse (precision) block matrix $\boldsymbol{\Sigma}_0^{-1}$ needs to satisfy $\boldsymbol{\Sigma}_0\boldsymbol{\Sigma}_0^{-1} = \mathbf{I}$. Given the defined structure of $\boldsymbol{\Sigma}_0^{-1}$, we can write:

$$\begin{bmatrix} \boldsymbol{\Sigma}_{tot} & \boldsymbol{\Sigma}_{ac} \\ \boldsymbol{\Sigma}_{ac} & \boldsymbol{\Sigma}_{tot} \end{bmatrix} \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}. \end{bmatrix} \tag{B.5}$$

Therefore, the blocks $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, and $\mathbf{D}$ must satisfy the following:

$$\boldsymbol{\Sigma}_{tot}\mathbf{A} + \boldsymbol{\Sigma}_{ac}\mathbf{C} = \mathbf{I}, \tag{B.6}$$
$$\boldsymbol{\Sigma}_{tot}\mathbf{B} + \boldsymbol{\Sigma}_{ac}\mathbf{D} = \mathbf{0}, \tag{B.7}$$
$$\boldsymbol{\Sigma}_{ac}\mathbf{A} + \boldsymbol{\Sigma}_{tot}\mathbf{C} = \mathbf{0}, \tag{B.8}$$
$$\boldsymbol{\Sigma}_{ac}\mathbf{B} + \boldsymbol{\Sigma}_{tot}\mathbf{D} = \mathbf{I}. \tag{B.9}$$

In order to solve the set of equations, we will start by expressing $\mathbf{A}$ from (B.8):

$$\mathbf{A} = -\boldsymbol{\Sigma}_{ac}^{-1}\boldsymbol{\Sigma}_{tot}\mathbf{C}. \tag{B.10}$$

Substituting it to (B.6) yields:

$$\begin{aligned} \mathbf{I} &= -\boldsymbol{\Sigma}_{tot}\boldsymbol{\Sigma}_{ac}^{-1}\boldsymbol{\Sigma}_{tot}\mathbf{C} + \boldsymbol{\Sigma}_{ac}\mathbf{C} \\ \mathbf{C} &= (\boldsymbol{\Sigma}_{ac} - \boldsymbol{\Sigma}_{tot}\boldsymbol{\Sigma}_{ac}^{-1}\boldsymbol{\Sigma}_{tot})^{-1}. \end{aligned} \tag{B.11}$$

Having obtained expression for $\mathbf{C}$, we can substitute it back to (B.10), which provides expression for $\mathbf{A}$:

$$\mathbf{A} = -\boldsymbol{\Sigma}_{ac}^{-1}\boldsymbol{\Sigma}_{tot}(\boldsymbol{\Sigma}_{ac} - \boldsymbol{\Sigma}_{tot}\boldsymbol{\Sigma}_{ac}^{-1}\boldsymbol{\Sigma}_{tot})^{-1}. \tag{B.12}$$

Following analogous procedure with (B.7) and (B.9) we obtain expression for the remaining unknowns:

$$\begin{aligned} \mathbf{B} &= -\boldsymbol{\Sigma}_{tot}^{-1}\boldsymbol{\Sigma}_{ac}(\boldsymbol{\Sigma}_{tot} - \boldsymbol{\Sigma}_{ac}\boldsymbol{\Sigma}_{tot}^{-1}\boldsymbol{\Sigma}_{ac})^{-1}, \\ \mathbf{D} &= (\boldsymbol{\Sigma}_{tot} - \boldsymbol{\Sigma}_{ac}\boldsymbol{\Sigma}_{tot}^{-1}\boldsymbol{\Sigma}_{ac})^{-1}. \end{aligned} \tag{B.13}$$

We have obtained expressions for all unknown blocks (matrices); therefore, we can invert the matrix $\boldsymbol{\Sigma}$. However, the expressions are rather complicated and contain multiple inversions. In the following text, we will seek simplification that will eventually allow for a less expensive computation. We observe that $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, and $\mathbf{D}$ contain the inverse of a term with a similar structure, which we will aim to simplify. First, let us focus on $\mathbf{A}$ and $\mathbf{C}$, containing the same term. Using the equality $\boldsymbol{\Sigma}_{tot} = \boldsymbol{\Sigma}_{ac} + \boldsymbol{\Sigma}_{wc}$, we can rewrite it as follows:

$$\begin{aligned} (\boldsymbol{\Sigma}_{ac} - \boldsymbol{\Sigma}_{tot}\boldsymbol{\Sigma}_{ac}^{-1}\boldsymbol{\Sigma}_{tot})^{-1} &= (\boldsymbol{\Sigma}_{ac} - (\boldsymbol{\Sigma}_{ac} + \boldsymbol{\Sigma}_{wc})\boldsymbol{\Sigma}_{ac}^{-1}(\boldsymbol{\Sigma}_{ac} + \boldsymbol{\Sigma}_{wc}))^{-1} \\ &= (-2\boldsymbol{\Sigma}_{wc} - \boldsymbol{\Sigma}_{wc}\boldsymbol{\Sigma}_{ac}^{-1}\boldsymbol{\Sigma}_{wc})^{-1} \\ &= (-\boldsymbol{\Sigma}_{wc}(2\mathbf{I} + \boldsymbol{\Sigma}_{ac}^{-1}\boldsymbol{\Sigma}_{wc}))^{-1} \\ &= -(2\mathbf{I} + \boldsymbol{\Sigma}_{ac}^{-1}\boldsymbol{\Sigma}_{wc})^{-1}\boldsymbol{\Sigma}_{wc}^{-1}. \end{aligned} \tag{B.14}$$

Using the Woodbury matrix identity, we can further manipulate the first inverted term:

$$(2\mathbf{I} + \boldsymbol{\Sigma}_{ac}^{-1}\boldsymbol{\Sigma}_{wc})^{-1} = \frac{1}{2}\mathbf{I} - \frac{1}{2}\boldsymbol{\Sigma}_{ac}^{-1}(\mathbf{I} + \boldsymbol{\Sigma}_{wc}\frac{1}{2}\boldsymbol{\Sigma}_{ac}^{-1})^{-1}\boldsymbol{\Sigma}_{wc}\frac{1}{2}. \tag{B.15}$$

Substituting it back, we eventually get

$$\begin{aligned} \left[-\frac{1}{2}\mathbf{I} + \frac{1}{4}\boldsymbol{\Sigma}_{ac}^{-1}(\mathbf{I} + \frac{1}{2}\boldsymbol{\Sigma}_{wc}\boldsymbol{\Sigma}_{ac}^{-1})^{-1}\boldsymbol{\Sigma}_{wc}\right]\boldsymbol{\Sigma}_{wc}^{-1} &= -\frac{1}{2}\boldsymbol{\Sigma}_{wc}^{-1} + \frac{1}{4}\boldsymbol{\Sigma}_{ac}^{-1}\left(\mathbf{I} + \frac{1}{2}\boldsymbol{\Sigma}_{wc}\boldsymbol{\Sigma}_{ac}^{-1}\right)^{-1} \\ &= -\frac{1}{2}\boldsymbol{\Sigma}_{wc}^{-1} + \frac{1}{4}\left(\boldsymbol{\Sigma}_{ac} + \frac{1}{2}\boldsymbol{\Sigma}_{wc}\right)^{-1}, \end{aligned} \tag{B.16}$$

which is simpler compared to the initial expression. Recalling $\mathbf{A}$, $\mathbf{C}$, and our result, we can rewrite them as

$$\mathbf{A} = \boldsymbol{\Sigma}_{ac}^{-1}\boldsymbol{\Sigma}_{tot}\left(-\frac{1}{2}\boldsymbol{\Sigma}_{wc}^{-1} + \frac{1}{4}\left(\boldsymbol{\Sigma}_{ac} + \frac{1}{2}\boldsymbol{\Sigma}_{wc}\right)^{-1}\right), \tag{B.17}$$

$$\mathbf{C} = \mathbf{B} = -\frac{1}{2}\boldsymbol{\Sigma}_{wc}^{-1} + \frac{1}{4}\left(\boldsymbol{\Sigma}_{ac} + \frac{1}{2}\boldsymbol{\Sigma}_{wc}\right)^{-1}. \tag{B.18}$$

An analogous procedure can be followed for the term common to $\mathbf{B}$ and $\mathbf{D}$. In this case, we make use of the fact that $\boldsymbol{\Sigma}_{ac} = \boldsymbol{\Sigma}_{tot} - \boldsymbol{\Sigma}_{wc}$:

$$\begin{aligned}(\boldsymbol{\Sigma}_{tot} - \boldsymbol{\Sigma}_{ac}\boldsymbol{\Sigma}_{tot}^{-1}\boldsymbol{\Sigma}_{ac})^{-1} &= (\boldsymbol{\Sigma}_{tot} - (\boldsymbol{\Sigma}_{tot} - \boldsymbol{\Sigma}_{wc})\boldsymbol{\Sigma}_{tot}^{-1}(\boldsymbol{\Sigma}_{tot} - \boldsymbol{\Sigma}_{wc}))^{-1} \\ &= (2\boldsymbol{\Sigma}_{wc} - \boldsymbol{\Sigma}_{wc}\boldsymbol{\Sigma}_{tot}^{-1}\boldsymbol{\Sigma}_{wc})^{-1} \\ &= (-\boldsymbol{\Sigma}_{wc}(-2\mathbf{I} + \boldsymbol{\Sigma}_{tot}^{-1}\boldsymbol{\Sigma}_{wc}))^{-1} \\ &= -(-2\mathbf{I} + \boldsymbol{\Sigma}_{tot}^{-1}\boldsymbol{\Sigma}_{wc})^{-1}\boldsymbol{\Sigma}_{wc}^{-1}.\end{aligned} \tag{B.19}$$

Employing the Woodbury matrix identity again, we get

$$(-2\mathbf{I} + \boldsymbol{\Sigma}_{tot}^{-1}\boldsymbol{\Sigma}_{wc})^{-1} = -\frac{1}{2}\mathbf{I} - \frac{-1}{2}\boldsymbol{\Sigma}_{tot}^{-1}(\mathbf{I} + \boldsymbol{\Sigma}_{wc}\frac{-1}{2}\boldsymbol{\Sigma}_{tot}^{-1})^{-1}\boldsymbol{\Sigma}_{wc}\frac{-1}{2}. \tag{B.20}$$

Substitution back will lead to the final expression:

$$\begin{aligned}-\left[-\frac{1}{2}\mathbf{I} - \frac{1}{4}\boldsymbol{\Sigma}_{tot}^{-1}(\mathbf{I} - \frac{1}{2}\boldsymbol{\Sigma}_{wc}\boldsymbol{\Sigma}_{tot}^{-1})^{-1}\boldsymbol{\Sigma}_{wc}\right]\boldsymbol{\Sigma}_{wc}^{-1} &= \frac{1}{2}\boldsymbol{\Sigma}_{wc}^{-1} + \frac{1}{4}\boldsymbol{\Sigma}_{tot}^{-1}\left(\mathbf{I} - \frac{1}{2}\boldsymbol{\Sigma}_{wc}\boldsymbol{\Sigma}_{tot}^{-1}\right)^{-1} \\ &= \frac{1}{2}\boldsymbol{\Sigma}_{wc}^{-1} + \frac{1}{4}\left(\boldsymbol{\Sigma}_{ac} + \frac{1}{2}\boldsymbol{\Sigma}_{wc}\right)^{-1}.\end{aligned} \tag{B.21}$$

Finally, we can rewrite $\mathbf{B}$ and $\mathbf{D}$ using our result:

$$\mathbf{B} = -\boldsymbol{\Sigma}_{tot}^{-1}\boldsymbol{\Sigma}_{ac}\left(\frac{1}{2}\boldsymbol{\Sigma}_{wc}^{-1} + \frac{1}{4}\left(\boldsymbol{\Sigma}_{ac} + \frac{1}{2}\boldsymbol{\Sigma}_{wc}\right)^{-1}\right), \tag{B.22}$$

$$\mathbf{D} = \mathbf{A} = \frac{1}{2}\boldsymbol{\Sigma}_{wc}^{-1} + \frac{1}{4}\left(\boldsymbol{\Sigma}_{ac} + \frac{1}{2}\boldsymbol{\Sigma}_{wc}\right)^{-1}. \tag{B.23}$$

By straightforward manipulations of our expressions for $\mathbf{A}$ and $\mathbf{D}$, we realize that they are identical. The same holds also for $\mathbf{B}$ and $\mathbf{C}$. Therefore, we can always use a simpler expression of the two. Finally, we can formulate a simplified version of the covariance matrix inverse as

$$\boxed{\boldsymbol{\Sigma}^{-1} = \begin{bmatrix}\mathbf{D} & \mathbf{C} \\ \mathbf{C} & \mathbf{D}\end{bmatrix} = \frac{1}{2}\begin{bmatrix}\boldsymbol{\Sigma}_{wc}^{-1} + (2\boldsymbol{\Sigma}_{ac} + \boldsymbol{\Sigma}_{wc})^{-1} & -\boldsymbol{\Sigma}_{wc}^{-1} + (2\boldsymbol{\Sigma}_{ac} + \boldsymbol{\Sigma}_{wc})^{-1} \\ -\boldsymbol{\Sigma}_{wc}^{-1} + (2\boldsymbol{\Sigma}_{ac} + \boldsymbol{\Sigma}_{wc})^{-1} & \boldsymbol{\Sigma}_{wc}^{-1} + (2\boldsymbol{\Sigma}_{ac} + \boldsymbol{\Sigma}_{wc})^{-1}\end{bmatrix}.} \tag{B.24}$$

## B.3 Score Computation with the Simplified PLDA

It was shown in Section 2.4.2, (2.43) that when using the simplified PLDA model, a verification score for a pair of embeddings $\boldsymbol{\varphi}_1$ and $\boldsymbol{\varphi}_2$ can be computed as follows:

$$\begin{aligned}s(\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2) = &-\frac{1}{2}\log|\boldsymbol{\Sigma}_0| - \frac{1}{2}\left(\begin{bmatrix}\boldsymbol{\varphi}_1 \\ \boldsymbol{\varphi}_2\end{bmatrix} - \begin{bmatrix}\mathbf{m} \\ \mathbf{m}\end{bmatrix}\right)^{\mathsf{T}}\boldsymbol{\Sigma}_0^{-1}\left(\begin{bmatrix}\boldsymbol{\varphi}_1 \\ \boldsymbol{\varphi}_2\end{bmatrix} - \begin{bmatrix}\mathbf{m} \\ \mathbf{m}\end{bmatrix}\right) \\ &+\frac{1}{2}\log|\boldsymbol{\Sigma}_1| + \frac{1}{2}\left(\begin{bmatrix}\boldsymbol{\varphi}_1 \\ \boldsymbol{\varphi}_2\end{bmatrix} - \begin{bmatrix}\mathbf{m} \\ \mathbf{m}\end{bmatrix}\right)^{\mathsf{T}}\boldsymbol{\Sigma}_1^{-1}\left(\begin{bmatrix}\boldsymbol{\varphi}_1 \\ \boldsymbol{\varphi}_2\end{bmatrix} - \begin{bmatrix}\mathbf{m} \\ \mathbf{m}\end{bmatrix}\right),\end{aligned} \tag{B.25}$$

where the mean $\mathbf{m}$ is defined by the model in (2.39), covariance matrices $\mathbf{\Sigma}_0$ and $\mathbf{\Sigma}_1$ were introduced in (B.2). For completeness, we recall that

$$\mathbf{\Sigma}_0 = \begin{bmatrix} \mathbf{\Sigma}_{tot} & \mathbf{\Sigma}_{ac} \\ \mathbf{\Sigma}_{ac} & \mathbf{\Sigma}_{tot} \end{bmatrix}, \qquad \mathbf{\Sigma}_1 = \begin{bmatrix} \mathbf{\Sigma}_{tot} & \mathbf{0} \\ \mathbf{0} & \mathbf{\Sigma}_{tot} \end{bmatrix}. \tag{B.26}$$

The inverse matrix $\mathbf{\Sigma}_0^{-1}$ was provided in (B.24). Inverting $\mathbf{\Sigma}_1$ is straightforward:

$$\mathbf{\Sigma}_1^{-1} = \begin{bmatrix} \mathbf{\Sigma}_{tot}^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{\Sigma}_{tot}^{-1} \end{bmatrix}. \tag{B.27}$$

Expanding (B.25), we get

$$\begin{aligned} s(\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2) = &-\frac{1}{2}\log|\mathbf{\Sigma}_0| - \frac{1}{2}(\boldsymbol{\varphi}_1^\intercal \mathbf{D}\boldsymbol{\varphi}_1 + \boldsymbol{\varphi}_2^\intercal \mathbf{C}\boldsymbol{\varphi}_1 + \boldsymbol{\varphi}_1^\intercal \mathbf{C}\boldsymbol{\varphi}_2 + \boldsymbol{\varphi}_2^\intercal \mathbf{D}\boldsymbol{\varphi}_2) \\ &+ (\boldsymbol{\varphi}_1^\intercal \mathbf{D}\mathbf{m} + \boldsymbol{\varphi}_1^\intercal \mathbf{C}\mathbf{m} + \boldsymbol{\varphi}_2^\intercal \mathbf{C}\mathbf{m} + \boldsymbol{\varphi}_2^\intercal \mathbf{D}\mathbf{m}) \\ &- (\mathbf{m}^\intercal \mathbf{D}\mathbf{m} + \mathbf{m}^\intercal \mathbf{C}\mathbf{m}) \\ &+ \frac{1}{2}\log|\mathbf{\Sigma}_1| + \frac{1}{2}(\boldsymbol{\varphi}_1^\intercal \mathbf{\Sigma}_{tot}^{-1}\boldsymbol{\varphi}_1 + \boldsymbol{\varphi}_2^\intercal \mathbf{\Sigma}_{tot}^{-1}\boldsymbol{\varphi}_2) \\ &- (\boldsymbol{\varphi}_1^\intercal \mathbf{\Sigma}_{tot}^{-1}\mathbf{m} + \boldsymbol{\varphi}_2^\intercal \mathbf{\Sigma}_{tot}^{-1}\mathbf{m}) + \mathbf{m}^\intercal \mathbf{\Sigma}_{tot}^{-1}\mathbf{m}, \end{aligned} \tag{B.28}$$

which can be further rewritten as

$$\begin{aligned} s(\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2) = &-\frac{1}{2}\log|\mathbf{\Sigma}_0| + \frac{1}{2}\log|\mathbf{\Sigma}_1| + \boldsymbol{\varphi}_1^\intercal \left(-\frac{1}{2}\mathbf{C}\right)\boldsymbol{\varphi}_2 + \boldsymbol{\varphi}_2^\intercal \left(-\frac{1}{2}\mathbf{C}\right)\boldsymbol{\varphi}_1 \\ &+ \boldsymbol{\varphi}_1^\intercal \left(-\frac{1}{2}\mathbf{D} + \frac{1}{2}\mathbf{\Sigma}_{tot}^{-1}\right)\boldsymbol{\varphi}_1 + \boldsymbol{\varphi}_2^\intercal \left(-\frac{1}{2}\mathbf{D} + \frac{1}{2}\mathbf{\Sigma}_{tot}^{-1}\right)\boldsymbol{\varphi}_2 \\ &+ (\boldsymbol{\varphi}_1 + \boldsymbol{\varphi}_2)^\intercal(\mathbf{D} + \mathbf{C} - \mathbf{\Sigma}_{tot}^{-1})\mathbf{m} + \mathbf{m}^\intercal(-\mathbf{D} - \mathbf{C} + \mathbf{\Sigma}_{tot}^{-1})\mathbf{m}. \end{aligned} \tag{B.29}$$

We can now recognize the repeating terms. By giving them names and grouping constant terms, we eventually obtain

$$\boxed{s(\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2) = \boldsymbol{\varphi}_1^\intercal \mathbf{\Lambda}\boldsymbol{\varphi}_2 + \boldsymbol{\varphi}_2^\intercal \mathbf{\Lambda}\boldsymbol{\varphi}_1 + \boldsymbol{\varphi}_1^\intercal \mathbf{\Gamma}\boldsymbol{\varphi}_1 + \boldsymbol{\varphi}_2^\intercal \mathbf{\Gamma}\boldsymbol{\varphi}_2 + (\boldsymbol{\varphi}_1 + \boldsymbol{\varphi}_2)^\intercal \mathbf{c} + k,} \tag{B.30}$$

where

$$\mathbf{\Lambda} = \frac{1}{4}\mathbf{\Sigma}_{wc}^{-1} - \frac{1}{4}(2\mathbf{\Sigma}_{ac} + \mathbf{\Sigma}_{wc})^{-1} \tag{B.31}$$

$$\mathbf{\Gamma} = -\frac{1}{4}(2\mathbf{\Sigma}_{ac} + \mathbf{\Sigma}_{wc})^{-1} - \frac{1}{4}\mathbf{\Sigma}_{wc}^{-1} + \frac{1}{2}\mathbf{\Sigma}_{tot}^{-1} \tag{B.32}$$

$$\mathbf{c} = ((2\mathbf{\Sigma}_{ac} + \mathbf{\Sigma}_{wc})^{-1} - \mathbf{\Sigma}_{tot}^{-1})\mathbf{m} \tag{B.33}$$

$$k = \log|\mathbf{\Sigma}_{tot}| - \frac{1}{2}\log|\mathbf{\Sigma}_{wc}| - \frac{1}{2}\log|2\mathbf{\Sigma}_{ac} + \mathbf{\Sigma}_{wc}| - \mathbf{m}^\intercal \mathbf{c}. \tag{B.34}$$

We note that we made use of the following equalities:

$$|\mathbf{\Sigma}_0| = \left|\begin{bmatrix} \mathbf{\Sigma}_{tot} & \mathbf{\Sigma}_{ac} \\ \mathbf{\Sigma}_{ac} & \mathbf{\Sigma}_{tot} \end{bmatrix}\right| = |\mathbf{\Sigma}_{tot} - \mathbf{\Sigma}_{ac}| \cdot |\mathbf{\Sigma}_{tot} + \mathbf{\Sigma}_{ac}| = |\mathbf{\Sigma}_{wc}| \cdot |2\mathbf{\Sigma}_{ac} + \mathbf{\Sigma}_{wc}|, \tag{B.35}$$

$$|\mathbf{\Sigma}_1| = \left|\begin{bmatrix} \mathbf{\Sigma}_{tot} & \mathbf{0} \\ \mathbf{0} & \mathbf{\Sigma}_{tot} \end{bmatrix}\right| = |\mathbf{\Sigma}_{tot}| \cdot |\mathbf{\Sigma}_{tot}|. \tag{B.36}$$