Deliverable D3.6: Audio-visual speaker separation/tracking with a moving robot

Due Date: 30/05/2024

Main Author: Sharon Gannot (BIU)

Contributors: -

Dissemination: Public Deliverable

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement No. 871245.
DOCUMENT FACTSHEET

<table>
<thead>
<tr>
<th>Deliverable</th>
<th>D3.6: Audio-visual speaker separation/tracking with a moving robot</th>
</tr>
</thead>
<tbody>
<tr>
<td>Responsible Partner</td>
<td>BIU</td>
</tr>
<tr>
<td>Work Package</td>
<td>WP3: Robust Audio-visual Perception of Humans</td>
</tr>
<tr>
<td>Task</td>
<td>T3.3: Extraction of Desired Sources (Moving Robot)</td>
</tr>
<tr>
<td>Version &amp; Date</td>
<td>30/05/2024</td>
</tr>
<tr>
<td>Dissemination</td>
<td>Public Deliverable</td>
</tr>
</tbody>
</table>

CONTRIBUTORS AND HISTORY

<table>
<thead>
<tr>
<th>Version</th>
<th>Editor</th>
<th>Date</th>
<th>Change Log</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BIU</td>
<td>30/05/2024</td>
<td>Final Draft</td>
</tr>
</tbody>
</table>

APPROVALS

<table>
<thead>
<tr>
<th>Authors/editors</th>
<th>Sharon Gannot (BIU)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task Leader</td>
<td>BIU</td>
</tr>
<tr>
<td>WP Leader</td>
<td>BIU</td>
</tr>
</tbody>
</table>
Contents

Executive Summary 3

1 Introduction 5

2 Literature Survey 6
   2.1 Single-Microphone Speaker Separation 6
   2.2 Concurrent Speaker Detector 6

3 Audio Pipeline 8
   3.1 Single-Speaker Track 8
   3.2 Dual-Speaker Track 8
   3.3 Arbiter 9

4 Single-Microphone Speaker Separation and Voice Activity Detection in Noisy and Reverberant Environments 10
   4.1 Problem Formulation 10
   4.2 Proposed Model 11
      4.2.1 Separation Module 11
      4.2.2 Online Mode 13
      4.2.3 VAD network 13
      4.2.4 Objective Functions 13

5 Concurrent Speaker Detector 14
   5.1 Problem Formulation 14
   5.2 Proposed Model 14
      5.2.1 Pre-Processing and Input Features 15
      5.2.2 Architecture 15
      5.2.3 Single- and multichannel Embedding Blocks 15
      5.2.4 Objective Functions 16
      5.2.5 Confidence Calibration 16
   5.3 Concurrent Speaker Detector (CSD): Simulation Study with Public Databases 17
      5.3.1 Databases 17
      5.3.2 Algorithm Setup 17
      5.3.3 Results 17

6 Audio Pipeline: Experimental Study using Lab Recordings 18
   6.1 Static Robot: Experiments and Evaluation of the Separation Algorithm 18
      6.1.1 Setup 18
      6.1.2 WER Results 18
   6.2 Moving Robot: Experiments and Evaluation of the Separation, Noise Reduction, Speaker ID, and Diarisation Modules 19
      6.2.1 Experimental Setup 19
      6.2.2 Audio Separation Track ('Red' System) 21
      6.2.3 Noise Reduction and Diarisation Track ('Blue' System) 22

7 Conclusions 25

Bibliography 26
Executive Summary

Deliverable 3.6 reports the progress on task T3.3 on audio speaker diarisation and separation with a moving robot, which is part of WP3: Robust Audio-visual Perception of Humans. The goal of task 3.3 is to provide several separated audio streams to be transcribed by the automatic speech recognition (ASR) and fed to the multi-party conversational system that will be deployed on ARI, the robotic platform designed by PAL Robotics for the SPRING project.

In this report, we describe experiments carried out at the BIU acoustic lab. We report on two dynamic scenarios:

1. Only a single speaker is active at each time segment, but several speakers can be non-concurrently active during the same utterance and should be transcribed into two text strings.

2. Two speakers are active with partial overlap. Hence, the raw audio should be separated into two audio streams, which should consequently be transcribed by the ASR.

We will describe the audio pipeline with its two parallel audio-processing tracks. We will also detail two algorithms: 1) single microphone separation (an updated version of the algorithm described in D3.4) and 2) concurrent speaker detector.
1 Introduction

This deliverable is part of WP3 of the H2020 SPRING project. The objective of WP3 is “the robust extraction, from the raw auditory and visual data, of users’ low-level characteristics, namely: position, speaking status and speech signal.” Following this objective, WP3 has two main outcomes:

1. The Multi-Person Tracking module, jointly exploiting auditory and visual raw data to detect, localise and track multiple speakers (corresponds to T3.1).
2. The Diarisation and Separation and the Speech Recognition modules, extracting the desired speaker(s) from a speech dynamic mixture and recognising the speech utterances from the separated sources, for a static T3.2 and a moving T3.3 robot.

This document focuses on the audio-processing modules responsible for noise reduction, diarisation, and speaker separation, together with the associated algorithms. The audio-visual speaker tracking module was described in D3.5.

We also describe in detail a recently developed algorithm for determining the speakers’ activity in the acoustic scene, which is used to control the audio pipeline and to provide the transcribed text and the associated speaker ID.

**Single microphone noise reduction and diarisation:** We apply the noise reduction algorithm (MoDE [6], already reported in previous deliverables) and a speaker diarisation scheme utilising the ECAPA2 Speaker ID package.

**Single-microphone speaker separation:** This algorithm applies a temporal convolutional network (TCN) module for separating the sources. The encoder and decoder are implemented as short-time Fourier transform (STFT) and inverse short-time Fourier transform (iSTFT). Simultaneously, the algorithm infers the activity patterns of the speakers. This algorithm was already reported in D3.4.\(^1\) Here we briefly summarize the algorithm, including recent improvements, and report on new results obtained by the moving robot.

We also discuss in detail the entire audio pipeline, as formed in the final stages of the project. The audio pipeline comprises two main tracks, one for non-overlapping speakers and the other for (partially) overlapping speakers. An “Arbiter” selects between the outputs of the track based on the current speakers’ activities. The audio pipeline also includes ASR, RIVA 2.7 by NVIDIA, and a simple direction-of-arrival (DOA) estimation (see Deliverable 3.5) that is later associated with the visual speaker localization readings. The transcribed speech utterance(s) and the active speakers’ IDs are published for further processing by the dialogue system.

---

\(^1\)Available at https://gitlab.inria.fr/spring/wp3_av_perception/audio_separation
2 Literature Survey

2.1 Single-Microphone Speaker Separation

Blind speaker separation is a rapidly growing research field aiming to separate a mixture of speech signals into their components. This technology has important applications in robot audition, speech recognition, hearing aids, and telecommunications. The introduction of deep clustering (DC) [18] and permutation invariant training (PIT) [48] has sparked increasing interest in single-microphone speaker separation algorithms that rely on deep neural networks (DNNs).

Many recent separation schemes are directly applied in the time domain using learnable encoder-decoder structure, e.g., Conv-Tasnet [29], successive downsampling and resampling of multi-resolution features (SuDoRmRf) [39], MossFormer [52], SepFormer [37], dual-path recurrent neural network (DPRNN) [28] and more [30, 33, 50].

The Conv-Tasnet algorithm employs one-dimensional convolutional layers to infer self-learned representations of the input signal. These embedded-domain representations are utilized to compute masks for each individual speaker, which are subsequently applied to separate the speakers. This approach employs the scale-invariant signal-to-distortion ratio (SI-SDR) [24] loss function, specifically designed to capitalize on temporal information.

The SuDoRmRf, presented in [39], is based on simple one-dimensional convolution layers requiring significantly less memory and parameters than the other methods listed above. These approaches mainly focused on a synthetic dataset, such as WSJ-2mix [18], WHAM! [45] and WHAMR! [31].

Several challenges and limitations remain unresolved in the methods mentioned above. In the presence of background noise and high reverberation, the quality of the separated signals is degraded. Robustness to noise and reverberation is a key challenge in real-world applications of speech source separation. Specifically, the frame length is relatively short in time domain approaches, allowing proper separation only in low reverberation environments [7], [17].

An additional time domain networks are MossFormer [52] and MossFormer2 [53]. MossFormer is a gated single-head transformer architecture with convolution-augmented joint self-attentions. MossFormer employs a joint local and global self-attention architecture that simultaneously performs a full-computation self-attention on local chunks and a linearized low-cost self-attention over the full sequence. MossFormer2 is a hybrid Transformer and recurrent model for monaural speech separation based on MossFormer. MossFormer2 substantially increases model size to 55.7 million parameters.

TF-GridNet [43] is a DNN architecture integrating full-band and sub-band modeling techniques to capture both local and global spectro-temporal information. The model employs a combination of convolutional layers, bidirectional long short-term Memory (BLSTM) networks, and self-attention mechanisms to enhance the separation performance.

2.2 Concurrent Speaker Detector

Speaker detection, namely the ability to identify and track the activities of individual speakers in an audio stream, is an important task with many practical applications. In particular, concurrent speaker detector (CSD) is the problem of identifying speakers’ presence and overlapping activity in a given audio signal. It classifies audio segments into three classes, namely: 1) no speech activity (noise only), 2) only a single speaker is active, and 3) more than one speaker is active. A reliable CSD is a key component in audio scene analysis and speech processing applications, e.g., speech detection, speaker counting and diarization, and multi-microphone spatial processing in “cocktail party” scenarios. CSD is a challenging task due to the complex nature of human speech. Accent, pitch, and speaking style variations can make the identification and detection of the speakers’ activity challenging. Consequently, developing effective CSD approaches is an active area of research, aiming to improve accuracy and robustness.

In [5], a multichannel CSD, based on convolutional neural network (CNN) architecture, was used as a building block of a linearly constrained minimum variance (LCMV) beamformer for controlling the estimation of its components, based on the speakers’ activity patterns. Specifically, the spatial correlation matrix of the noise is estimated during noise-only segments, and the steering vectors of the beamformer are estimated during single active speaker seg-
ments. The beamformer’s weights are not updated during concurrent activities of more than one speaker. In [19, 47], both CNN and attention mechanisms are employed for speaker-counting and identification. In [2, 36], a long short-term memory (LSTM) model is used for the task of overlapped speech detector (OSD). Unlike the CSD, only two classes comprise the OSD task. The first comprises noise-only or single-speaker segments, while the second comprises overlapped speech segments, namely two or more concurrent speakers.

The Transformer model, which was originally proposed in the natural language processing (NLP) domain [14, 40], was later adopted by the audio community for various tasks, e.g., speech separation [38] and audio classification [15]. It was demonstrated in [15] that the audio spectrogram transformer (AST) model, which is an adaptation of the vision transformer (ViT) model [12], outperforms CNN-based models. We stress that the AST model only processes single-microphone data, whereas multiple microphones are available in many real-world use cases. It is well-known that, if properly utilized, the additional spatial information may improve performance. In [9], a model based on TCN is used, and in [10], a Transformer-based model is used. Both models estimate the activity of the speakers. Specifically, two related tasks are implemented, voice activity detector (VAD) and OSD, as well as their joint estimation. The VAD classifies audio segments into two classes: 1) no active speaker and 2) speech activity (either one or more speakers). We can therefore refer to the combined VAD and OSD task as a CSD task. A Transformer-based solution is also utilized for audio OSD [54] and for audio-visual OSD [23]. The CSD, which is a multi-class classification task, is more complex than OSD or VAD, which are binary classification tasks.
3 Audio Pipeline

The audio pipeline streamlines all relevant algorithms. It processes the raw audio signal(s), enhances it, separates the mixed speakers if necessary, and applies an ASR to transcribe the utterance for further processing by the dialogue system. In parallel, it infers the activity of the speakers, the identity of the speakers, and their DOAs. Figure 3.1 illustrates the audio pipeline, comprising two parallel processing tracks and an arbiter (depicted by the green blocks) to select between their outcomes. The upper track (depicted by the blue blocks) is responsible for single-speaker scenarios, and the lower track (depicted by the red blocks) for dual-speaker scenarios involving concurrent speakers. The two processing tracks are utilising a single microphone, besides the DOA estimation block that required at least two microphones to extract the spatial information.

3.1 Single-Speaker Track

The single-speaker processing track is responsible for noise reduction and diarisation of the speech utterance in cases of non-overlapping speakers. It follows a relatively straightforward flow. It takes the raw audio input and first applies the Mixture of Deep Experts (MoDE) module [6] to suppress the ambient noise. The goal of this stage is to improve the quality and clarity of the audio signal before subsequent steps.

Next, an ASR module is applied, which transcribes the audio signal to a text stream, which is sent to the dialogue module at the end of the utterance. We used NVIDIA RIVA 2.7 (based on CTC-Conformer architecture).\(^1\)

The output of the MoDE module is fed in parallel to the ‘Speaker ID’ module to relate the ASR transcription to a specific speaker in the current interaction of the robot and the humans.

Speaker identification is used to diarise the utterance. We first use a frame-based commercial speaker embedding package, ECAPA2.\(^2\) This module is applied to 0.5-second long speech segments only if a VAD is activated. We used a commercial VAD product, WebRTC.\(^3\) The embeddings are clustered online by calculating the cosine distance between the new embedding and the centroid of already acquired clusters. A kernel is applied to these distances to enhance the robustness of the classification results.

The sequence of Speaker ID readings is smoothed by a median filter with a buffer size of 9 to avoid outliers.

The diarisation (“who speaks when”) process associates the identified speaker labels (i.e., the clustering labels) with the corresponding transcribed text, thus enabling accurate attribution of spoken content to individual speakers, a crucial requirement for applications involving multi-talker interactions or conversations. After 0.8-1.5 sec of no active speech, the end of the sentence is declared, and the final ASR results, associated with the active Speaker ID, are published to the dialogue manager.

We are also currently examining an alternative framework for overlap-aware low-latency online speaker diarisation based on end-to-end local segmentation.\(^8\)

In parallel to the enhancement/transcription modules we apply DOA estimation to two microphone signals (this is the only multi-microphone processing in the audio pipeline). We use the classical generalized cross-correlation with phase transform (GCC-PHAT) method [20] with parabolic sub-sample interpolation. Note that the GCC-PHAT readings are only valid during single-speaker frames.

3.2 Dual-Speaker Track

The dual-speaker processing track addresses more complex scenarios where two speakers are actively speaking, potentially overlapping or interacting. It incorporates additional processing steps to handle this increased complexity effectively.

\(1\)https://docs.nvidia.com/deeplearning/riva/user-guide/docs/reference/models/asr.html
\(2\)https://huggingface.co/Jenthe/ECAPA2
\(3\)https://webrtc.org/
The first stage in this pipeline is again the VAD module that identifies the active frames in the audio stream. Following the VAD, a speech separation algorithm is applied (see Chap. 4. This crucial step allows the subsequent stages to process each speaker’s audio independently. With the speaker streams separated, the pipeline duplicates into two parallel branches, each applying an ASR client and a corresponding Speaker ID module. These components execute the same functions as their counterparts in the single-speaker track, but they operate on the individual separated speaker streams rather than on the combined audio.

![Full audio pipeline diagram](image)

**Figure 3.1: Full audio pipeline.**

### 3.3 Arbiter

The outcomes of both single-speaker and dual-speaker pipelines are fed to the Audio Arbiter module. The audio arbiter is responsible for choosing which transcription is the most accurate, as well as forwarding the DOA and speaker ID decisions. The arbiter uses the CSD module, explained in Chap. 5. The CSD determines the activities of the speakers, specifically if multiple speakers are concurrently active. In the latter case, the ‘red’ processing track will be selected.

We note that the final decision regarding the utterance is only taken after the ASR has declared ‘end-of-utterance’. This usually happens after a time segment, for which no speech activity has been detected, elapsed. The predefined length is usually set in the range of 0.8-1.5 sec. This delayed decision may allow us to apply offline processing of the entire utterance if required. In Chap. 6, we therefore examine both offline and online variants of the separation module.
4 Single-Microphone Speaker Separation and Voice Activity Detection in Noisy and Reverberant Environments

We propose the Separation TF Attention Network (Sep-TFAnet), a modified separation approach that leverages the Conv-Tasnet [29] backbone, using TCN, toward a more realistic scenario. By incorporating an attention block and training with reverberant references, the proposed network improves the separation scores in real-world challenging environments compared to other competing methods. In Sep-TFAnet we substitute the learnable time domain encoder-decoder with STFT-iSTFT analysis-synthesis modules to better deal with high reverberation scenarios. In [7], it was demonstrated that the STFT-iSTFT is a more valid choice for the encoder and decoder stages than the learnable representations in reverberant environments. It is also demonstrated that jointly addressing both reverberation and overlapped speech appears to be a challenging task and therefore we use the reverberated signals as our ground truth references.

In addition, we propose Sep-TFAnet\textsuperscript{VAD}, which concurrently infers all speakers’ activity patterns. In other words, together with the separation, we jointly train a VAD network. VAD is an essential component in multiple downstream tasks, e.g., post-filters, beamforming, and localization. Note that several papers combine VAD with multichannel speech processing [42], and for speaker extraction [26] [41].

One of the main objectives of Sep-TFAnet\textsuperscript{VAD} is to improve the auditory capabilities of a robot. Developing socially assistive robots that can perform multi-person interactions requires progress in speech-related tasks such as speech source separation, localization, etc. The robot is supposed to function in a crowded environment. Therefore, we trained and adapted our system to complex acoustic conditions. In this context, the VAD can be utilized as a dialogue controller, assisting the robot in engaging in human-like interactions. The VAD decisions can facilitate the application of a post-filtering operation to suppress further residual interfering signals and noise and enable the estimation of several building blocks of subsequent multichannel processing, e.g., an LCMV beamformer [4].

4.1 Problem Formulation

Let \( x(t) \) be a mixture of \( I \) concurrent speakers captured by a single microphone:

\[
x(t) = \sum_{i=1}^{I} \{s_i * h_i\}(t) + n(t) \quad t \in \{0, \ldots, T - 1\},
\]

where \( s_i(t) \) represents the signal of the \( i \)-th speaker, \( h_i(t) \) represents the room impulse response (RIR) between the \( i \)-th speaker and the microphone, and \( n(t) \) represents an additive noise. In this paper, we deal with static scenarios. In the STFT domain, and under the assumption of sufficiently long time frames, Eq. (4.1) can be reformulated as,

\[
x(l, k) = \sum_{i=1}^{I} s_i(l, k)h_i(k) + n(l, k),
\]

where \( l \in \{0, \ldots, L - 1\} \) and \( k \in \{0, \ldots, K - 1\} \) are the time-frame and the frequency-bin indexes, respectively, with \( L \) the total number of time-frames and \( K \) the total number of frequency bins. In our study, we only address the case \( I = 2 \). Denote the outputs of the separator system in the STFT domain as \( \hat{s}_1(l, k) \) and \( \hat{s}_2(l, k) \).

We assume a fully supervised setting, in which the goal is to infer a model that separates two output signals \( \hat{s}_1(t) \) and \( \hat{s}_2(t) \) from an unseen mixture \( x(t) \), by maximizing the SI-SDR between the separated signals and the reverberated clean signals, \( s_{\text{rev},i}(t) = \{s_i * h_i\}(t) \) and \( s_{\text{rev},i}^{\text{rev},i}(t) = \{s_i * h_i\}(t) \), respectively.

In addition, in one version of our system, Sep-TFAnet\textsuperscript{VAD}, the activity of each separated speaker per time frame is determined by applying a VAD network. This task also assumes a fully-supervised training.
This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement No. 871245.

Figure 4.1: Sep-TFAnet\textsuperscript{VAD} architecture. Learnable blocks are depicted in orange, and data blocks are in blue.

4.2 Proposed Model

The model comprises two main components: a separation module and a VAD module. The separation module is based on a temporal convolutional network (TCN) backbone [25]. Rather than a learned encoder and decoder, we use the STFT and the iSTFT. As demonstrated in [7, 44], audio processing algorithms that are based on STFT-iSTFT are advantageous in high reverberation as compared with learned encoder-decoder. The VAD module determines the activity patterns of each separated signal and can be useful in downstream tasks.

A block diagram of the entire system is depicted in Fig. 4.1. Next, we elaborate on the various components of the system.

4.2.1 Separation Module

The separation module is the main component of our proposed scheme. The various components of this module are depicted in Fig. 4.2.

An STFT first analyzes the raw audio signal. Then, the log-spectrum is calculated, followed by layer normalization (LN). The result of these operations constitutes the input to the separation module. The main processing module is an adapted TCN (dashed part in Fig. 4.2a). Originally, the TCN is a series of identical 1-D Conv blocks with increasing dilation factors and with zero-padding along the time dimension to maintain their integrity. The dilation factor enables capturing a sufficiently long temporal context of the speech signal. We stress that past STFT frames are also relevant for the separation task in high reverberation levels and should be considered. Here, the adapted TCN consists of three repeats of a stack of eight 1-D AttConv blocks, as proposed in [29] and depicted in Figs. 4.2a, 4.2b.

Unlike the original 1-D Conv blocks in the Conv-Tasnet algorithm [29], our 1-D AttConv blocks have only one output and do not feature an additional skip connection output from the output of each block to the overall TCN output (see Fig. 4.2b). We have decided to avoid this additional output since according to empirical evidence indicated in [35] and affirmed by our findings, they do not improve performance but rather increase the number of parameters.

For each 1-D AttConv block (see Fig. 4.2b), the input of the block, in\textsuperscript{1-D AttConv}, is summed to the output of the normalization layer to get its overall output, out\textsuperscript{1-D AttConv}. Additionally, since the input to our network is an STFT representation, the dilation factors were chosen to maintain as small as possible receptive field but long enough to address reverberation. As a suitable compromise, we set $d = (i \mod 4) + 1, 0 \leq i \leq 7$, where $d$ is the dilation factor and $i$ denotes the number of the 1-D AttConv block in each repeat as depicted in Fig. 4.2a. The first $1 \times 1$ Conv layer in each 1-D AttConv block has a kernel size 1. The number of filters is set to $F$, the number of frequency bins, to capture the frame-wise frequency patterns. In addition, following the $1 \times 1$ Conv layer, the D-Conv layer expounded upon in [29], is subsequently applied with $H$ output filters, $H > F$ with the purpose of achieving a richer representation while concurrently minimizing the number of parameters.

Another component contributing to the network's performance is the TF-attention block adopted from [51], which was proposed in the context of noise reduction. The TF-attention block is depicted in Fig. 4.2c. This block is positioned after the final $1 \times 1$ Conv layer of each 1-D AttConv block, followed by a normalization layer, to optimize further the network's ability to learn to recognize complex patterns in speech data. As depicted in Fig. 4.2c, the block
Figure 4.2: Separation Module: Architecture.

(a) Separation module
(b) 1-D AttConv
(c) time frequency (TF)-attention block
commences with average pooling layers on both the time and frequency dimensions, followed by $1 \times 1$ Conv layers and activation functions. These layers generate an attention mask, subsequently multiplied element-wise with the modified spectrogram of the input signal.

Finally, after applying all 1-D AttConv blocks, a pre-exponential unit (PReLU) activation function, along with an LN and a $1 \times 1$ Conv, is employed to estimate the masks, which are subsequently passed through a Sigmoid activation function, which output is confined to the interval $[0, 1]$ as depicted in Fig. 4.2. In conjunction with the STFT, this activation function preserves the audio scale of the output along the utterance and can be beneficial in online mode.

The estimated speakers’ signals are transformed back to the time domain by augmenting the masked spectrum with the noisy and mixed input signal phase and then applying the iSTFT.

### 4.2.2 Online Mode

Developing an online variant of the separation algorithm is vital in human-robot communication, as it may facilitate proper interaction between the robot and the human speaker. We employ a distinctive low-latency strategy instead of processing the complete batch in the offline mode. We divide the input into short segments comprising past, present, and future samples. We apply the separation operation to each segment by using a sliding window. To maintain consistency of the separated signals between all segments, we apply PIT with $L_1$ loss. We worked with an overlapping sliding window with a non-causal part. We would like to prove that despite our choice of TF-attention block in our network, which averages across the entire temporal dimension for each signal, Sep-TFAnet can perform well in an online manner.

### 4.2.3 VAD network

The purpose of the VAD network is to infer the activity patterns of the separated speakers. The inputs to the network are the extracted masks from the jointly trained separation module. The VAD network consists of a 1-D convolution layer with four filters, followed by a PReLU activation function and a normalization layer. The activity patterns of the corresponding speakers are finally obtained by applying a 1-D convolutional layer with one filter, followed by a hard threshold to obtain a binary activity decision for each frame. In addition, we modify the 1-D AttConv block in the separation module, adopted from [27], such that the output of the 1-D AttConv block is:

$$
\text{out}^{1-D \text{ AttConv}} = \text{LN}(\text{in}^{1-D \text{ AttConv}} + \text{LN}(\text{in}^{1-D \text{ AttConv}} + \text{out}^{\text{TF Att}}))
$$

where $\text{in}^{1-D \text{ AttConv}}$ is the input to the AttConv block, and $\text{out}^{\text{TF Att}}$ is the output of the TF-attention block within the AttConv block.

### 4.2.4 Objective Functions

We experimented with several separation objective functions to efficiently train the model and found that the SI-SDR loss yields the best perceptual improvement. The loss is defined between the output of the algorithm and the reverberant component, SI-SDR ($s^{\text{rev}}, \hat{s}$) where $s^{\text{rev}}$ and $\hat{s}$ represent the concatenations of $s_i^{\text{rev}}(t)$ and $s_i(t)$ across all samples of the utterance, respectively. To alleviate the permutation problem, common to separation problems, utterance-level permutation invariant training (uPIT) [21] was employed. We stress that the target signals during training were the reverberant signals, namely the anechoic signals convolved with the corresponding RIRs. Hence, the network focuses on the separation task and does not attempt to dereverberate the separated signals. While this may improve separation scores, the performance of ASR systems may deteriorate in high reverberation levels. If reverberation is an issue, a dereverberation module, e.g., [46] or [11], can be applied as a post-filter.

For training the VAD module, we have used the binary cross entropy (BCE) loss:

$$
\text{BCE} = - \sum_{i=0}^{I} \sum_{l=0}^{L-1} v_i(l) \log(p_i(l)) + (1 - v_i(l)) \log(1 - p_i(l)),
$$

where $p_i(l)$ is the output of the VAD network indicating the probability of speaker $i \in \{0, \ldots, I\}$ at time frame $l$ to be active, with $v_i(l) \in \{0, 1\}$ is the ground truth activity of speaker $i$ at time frame $l$. 

D3.6: Audio-visual speaker separation/tracking with a moving robot
5 Concurrent Speaker Detector

We propose an algorithm to solve the CSD task. Our contribution is threefold: 1) we extend the use of ViT and adapt it to the multi-microphone case, 2) we incorporate in the training process a re-weighting mechanism according to the importance of each class and further use calibration to improve the classification accuracy, and finally 3) similarly to [2,9,10,23,36,54], we evaluate the performance of the proposed model on AMI [3] and CHiME 5 [1] databases, and additionally on the recently introduced AliMeeting [49] database (in Chinese).

5.1 Problem Formulation

Let \( X_i(\ell, k), i = 1, \ldots, N \) represent the STFT of the microphone signals, where \( N \) is the number of microphones, \( \ell \) and \( k \) represent the frame index and the frequency index, respectively. The goal of a CSD algorithm is to classify each audio segment (either single-microphone or multi-microphone) into one of the three classes:

\[
\text{CSD}(\ell) = \begin{cases} 
\text{Class #0} & \text{Noise only} \\
\text{Class #1} & \text{Single-speaker activity} \\
\text{Class #2} & \text{Concurrent-speaker activity}
\end{cases}
\]  

(5.1)

The statistical characteristics of the audio segments may change according to the scenario. Multiple types of noise may exist for class '0' ('Noise-Only'). Class '1' ('Single-speaker activity') can be challenging due to the variability of human speech. Individuals may have different accents, speaking styles, and vocal characteristics, making it difficult for algorithms to identify them accurately. In class '2' ('Concurrent-speaker activity'), the different number of active speakers may result in diverse statistical properties. In addition, the presence of background noise or reverberation can further complicate the task. Consequently, developing robust and accurate CSD methods that can handle a wide range of input conditions becomes essential.

5.2 Proposed Model

The proposed CSD model is based on the ViT [12] architecture, which has achieved state-of-the-art performance on a variety of computer vision tasks. We have modified the original ViT architecture to better suit audio processing requirements, including the use of log-spectrum as input and the ability to handle both single-channel and multichannel audio. The input features are the log-magnitude of the STFT of the audio signals, denoted hereinafter log-spectrum.

The model consists of three main blocks: Embedding, Transformer, and Classification. The first block linearly projects the input data and generates the input tokens for the Transformer model. The second block is a multi-head attention (MHA) transformer block, consisting of self-attention layers, which can capture complex relations within its input data. The attention mechanism [40] allows the model to simultaneously focus on different parts of the input, enabling it to learn rich feature representations from the log-spectrum. Finally, several fully-connected layers are applied to map the learned features to the final output predictions.

Our starting point is, therefore, the ViT model, with the images substituted by the log-spectra and the RGB channels by the multi-microphone measurements. The multichannel model attends to different areas in the input to achieve the best classification results. We use a Cross-Entropy (CE) loss function for training, a common choice for classification tasks. In addition, we used Label-Smoothing (LS) [32] and the Cost-Sensitive (CS) loss function [13] as regularization techniques to improve the ability of the model to generalize to unknown data. The high-level architecture of the model is presented in Fig. 5.1.
5.2.1 Pre-Processing and Input Features

The microphone signals are first resampled to 16kHz (if the original sampling rate differs from the nominal one). Subsequently, the signals are analyzed by an STFT with Hann window of length 512 and 50% overlap. Finally, the log-spectrum is calculated, resulting in 257 frequency bins per frame.

The output labels are determined using the transcribed databases, with a resolution of 0.1 Sec and context frames 0.2 Sec long on both sides of the analyzed segment. Therefore, each audio segment, lasting 0.5 seconds, is categorized into one of the three classes. The overall dimensions of the input tensor are $N \times 257 \times 32$, where $N$ is the number of microphones, 257 is the number of frequency bins, and 32 is the number of time bins.

5.2.2 Architecture

We will now elaborate on the three blocks comprising the model’s architecture.

The Embedding block receives the input log spectrum and produces the tokens utilized by the Transformer block. This block splits the input data into patches and linearly projects each to form tokens. We set the dimensions of a 2-D learnable kernel and strides values in the TF axes, such that each patch is projected to an embedding space with a dimension $D$ and considered a token. This process results in a features tensor with dimensions $\#$Tokens $\times D$. This block can be designed to handle both multichannel and single-channel signals, as will be explained below. Following [22], we apply a dual patch-normalization for improving the ViT results (see the ‘Normalization’ block in Fig. 5.1).

The Transformer block follows the ViT architecture with minor modifications. The Transformer was initialized with random weights and trained with the chosen databases. This module consists of several layers of MHA blocks. We followed the ViT architecture and used Class token [CLS], an additional learnable token added as an input to the Transformer. In addition, a learnable positional embedding is added to each input token before the first MHA layer. In total, the input and output of each MHA blocks are tokens of shape $(\#$Tokens $+ 1) \times D$.

The last block in our model, the Classification block, consists of two fully connected layers that map the Transformer output to fit the number of classes. The Classification block takes only the token corresponding to [CLS] as input. This should make the classification process unbiased towards any particular token, as discussed in [12].

5.2.3 Single- and multichannel Embedding Blocks

The Embedding block transforms the input data into tokens used by the Transformer block, comprising 12 Transformers. We stress that the AST model [15] addresses a different classification task and is limited to single-channel inputs. Since we are also interested in the multichannel case, the standard embedding block should be modified accordingly.

The basic, single-microphone structure, i.e. $N = 1$, is depicted in Fig. 5.2a. The input log spectrum is split into patches with a 2-D learnable kernel with a stride set to 1. In ViT, the shape of the patches is $16 \times 16$, but according to our analysis, a more useful patch size is $257 \times 8$, as it jointly analyzes the entire frequency axis. Later on, each patch is linearly projected to a dimension of $D = 768$, resulting in a tensor of shape $\#$Patches $\times 768$.

The information must be merged from the different channels for the multichannel case. The overall proposed structure is depicted in Fig. 5.2b. The most effective merging technique, denoted here as Type #1, entails independently applying a single-channel embedding to each microphone signal and then combining all channels through concatenation. This process yields an output tensor with dimensions $N \cdot \#$Patches $\times 768$. Ensuring identical channel numbers during both the training and testing stages is necessary for this structure. Furthermore, due to the expansion of the channel dimension compared to the single-microphone scenario, there is an increase in the number of input tokens for the subsequent Transformer block. Similarly, the input feature vector to the Classification block also increases. All of which increase the total number of parameters.

To further analyze the merging strategies, we have examined two alternatives, designated hereinafter Type #2 and Type #3.
In Type #2, each single-channel embedding block is independently applied, and a summation operation merges the information. The resulting data shape is \( \#\text{Patches} \times 768 \). While the independent processing of each channel may be beneficial performance-wise, it also requires the number of channels to be identical in the training and test stages.

In Type #3, the weights of all single-channel embedding blocks are shared (Siamese networks), and their output is then merged using an averaging operation. The shape of the result is again \( \#\text{Patches} \times 768 \). This structure is indifferent to a mismatch between the number of microphones in the training and test stages and simultaneously reduces the number of parameters. Nevertheless, it may fall short of fully capturing the relationships between the signals from the microphones.

After experimenting with all three alternatives, we decided to use Type #1 due to its ability to perform cross-channel attention, a property that enhances the overall performance of the proposed method. Table 5.1 compares the mean average precision (mAP) results for the three merging types for all databases. Table 5.1 further supports choosing Type #1 since it outperforms all three types for the OSD task while exhibiting only marginal performance degradation for the VAD task.

Table 5.1: Ablation study for merging strategies: The mean average precision (mAP) (%) measure for the VAD and the OSD classifiers, as well as the number of required parameters (#P) in millions (M).

<table>
<thead>
<tr>
<th></th>
<th>AMI VAD</th>
<th>AMI OSD</th>
<th>AMI #P (M)</th>
<th>AllMeeting VAD</th>
<th>AllMeeting OSD</th>
<th>AllMeeting #P (M)</th>
<th>CHIME VAD</th>
<th>CHIME OSD</th>
<th>CHIME #P (M)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type #1</td>
<td>98.2</td>
<td>98.1</td>
<td>98.1</td>
<td>98.2</td>
<td>87.8</td>
<td>98.1M</td>
<td>91.6</td>
<td>83.5</td>
<td>91.8</td>
</tr>
<tr>
<td>Type #2</td>
<td>98.1</td>
<td>69.5</td>
<td>98.1</td>
<td>99.6</td>
<td>73.8</td>
<td>98.1M</td>
<td>96.6</td>
<td>56.5</td>
<td>91.7</td>
</tr>
<tr>
<td>Type #3</td>
<td>97.9</td>
<td>71.9</td>
<td>86.9</td>
<td>98.3</td>
<td>86.4</td>
<td>86.9M</td>
<td>91.9</td>
<td>63.4</td>
<td>86.9</td>
</tr>
</tbody>
</table>

5.2.4 Objective Functions

As we aim at the classification task, the natural choice for a loss function is the Cross-Entropy (CE). However, the classification results were not balanced between the different classes in our databases. We have, therefore, used two additional methods to guide the model to focus on the more challenging examples in the data, mainly Class #1. We used Label-Smoothing (LS) [32] and class weights. Both demonstrated an improvement in classification accuracy. We also applied the Cost-Sensitive (CS) loss [13]. In this procedure, we follow a 2-stage training procedure: First, we train the model with no CS loss, then we modify the CS loss weights according to the results and retrain the model. The CS loss weights are defined as a \( 3 \times 3 \) matrix that gives more weight, i.e., increasing the loss function, for more frequent classification errors. When applying the CS loss, we use an extra hyper-parameter weighting between the CS loss and the CE loss. This hyper-parameter was tested extensively and was set to a value of 15-20 (depending on the different databases).

5.2.5 Confidence Calibration

After training the model, we applied confidence calibration using temperature scaling [16]. The model calibration can become handy when the model is used to control the estimation of the building blocks of a beamformer [5], where erroneous classification can significantly deteriorate the separation performance. Calibration enhances the interpretability of the model’s predictions by aligning them with probabilities, establishing an appropriate threshold for considering the prediction valid. An instance of utilizing a calibrated model involves setting segments with predictions
below a certain threshold to Class #2. These segments are then excluded from the estimation of the beamformer’s weights.

5.3 Concurrent Speaker Detector (CSD): Simulation Study with Public Databases

We report here on the results obtained using common databases.

5.3.1 Databases

We applied the proposed model to three real-world databases, namely AMI [3], AliMeeting [49], and CHiME 5 [1]. All databases use a microphone array for the recordings, AMI and AliMeeting use an 8-microphone array, and CHiME 5 uses a 4-microphone array. AMI database consists of 100 hours of meeting recordings with English speakers (both male and female) in 3 different rooms and setups. The AliMeeting comprises 118.75 hours of real meetings with 2-4 participants speaking in Mandarin. The CHiME 5 database consists of recordings of conversations between English speakers from different real-home environments. CHiME 5 has six arrays (U01 to U06) with four microphones each. We chose to train and report only for arrays U01 and U02.

5.3.2 Algorithm Setup

We used the architecture described in Section 5.2.3 with Type #1 Embedding block and CS loss for all models since, as discussed above, it stands out as the most effective scheme. In training the models, we used the Adam optimizer with a learning rate of $1 \times 10^{-6}$, a weight decay of $1 \times 10^{-9}$, and a batch size of 128. To prevent overfitting, considering the model’s substantial parameter count, we limit the number of epochs to a range of 10-15, depending on the examined database. The overall parameter count falls within the range of 86.9-98.1M, as illustrated in Table 5.1. We used the following hyperparameters: The Embedding block is set with a dimension of $D = 768$, and the Transformer block is set with 12 heads and a depth of 12. The classification block has one hidden layer with dimension 387.

5.3.3 Results

To gain insights into the proposed method’s performance and give a more detailed analysis of the errors, we chose to present the confusion matrix that compares the ground-truth labels with the predicted labels by our model (as a percentage normalized to the ground-truth labels). The confusion matrices for the single-microphone case are depicted in Table 5.2 and Table 5.3 for the multi-microphone case.

Table 5.2: CSD results: Single-microphone model confusion matrices, as [%] normalized to the ground-truth labels. ‘T’-true labels, ‘P’-predicted labels.

<table>
<thead>
<tr>
<th></th>
<th>AMI</th>
<th>AliMeeting</th>
<th>CHiME</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>T</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>78</td>
<td>20</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>9</td>
<td>75</td>
<td>16</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>37</td>
<td>62</td>
</tr>
</tbody>
</table>

Table 5.3: CSD results: Multi-microphone model confusion matrices, as [%] normalized to the ground-truth labels. Using the Type #1 embedding block. ‘T’-true labels, ‘P’-predicted labels.

<table>
<thead>
<tr>
<th></th>
<th>AMI</th>
<th>AliMeeting</th>
<th>CHiME</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>T</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>80</td>
<td>18</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>10</td>
<td>74</td>
<td>16</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>35</td>
<td>63</td>
</tr>
</tbody>
</table>

The multi-microphone results are slightly better. However, due to the unfavorable microphone constellation in ARI and the simplicity of the implementation, we decided to only implement the single-microphone version.
6 Audio Pipeline: Experimental Study using Lab Recordings

We carried out a recording campaign at the acoustic laboratory at Bar-Ilan University to evaluate the audio pipeline’s performance. The lab is a $6 \times 6 \times 2.4 \text{ m}$ room with a reverberation time controlled by 60 interchangeable panels covering the room facets. It allows for testing and analysis of the algorithms in a controlled environment under adverse acoustic conditions. In our experiments, the reverberation time was set (by changing the panel arrangements) to either 350 ms, typical of a meeting room, or 600 ms, typical of a lecture hall. The room layout is depicted in Fig. 6.1. In Sec. 6.1, we report on experiments with English utterances recorded by ARI in a static position and processed offline. This is an extension of the experimental study that was already reported in D3.4. Specifically, we add to the experimental study unbalanced scenarios in which one of the speakers is significantly weaker than the other.

In Sec. 6.2, we turn to a new set of experiments that specifically analyze the two audio processing tracks (the ‘blue’ and the ‘red’ modules). For this part, we also test the algorithms in a dynamic scenario with moving ARI.

6.1 Static Robot: Experiments and Evaluation of the Separation Algorithm

6.1.1 Setup

In our experimental setup, ARI was positioned at the center of the acoustic lab, with a set of loudspeakers in front of it, on two semi-circles with approximately 1 m and 2 m radius, respectively, as depicted in Fig. 6.1. Our experiments only used the inner semi-circle with five loudspeakers positioned at $[−65, −30, 0, 30, 65]^\circ$. To generate a sample, we randomly selected two loudspeakers and played speech utterances randomly drawn from the Librispeech test set [34]. The utterances were separately recorded by ARI and then manually mixed to enable SI-SDR calculation. The overlap between the speakers was uniformly sampled in the range $[25\%, 50\%]$. No external noise was added to the recordings; hence, only sensor and low-level ambient noise are present. Overall, 200 samples were generated at each reverberation level.

In certain scenarios, one speaker exhibits significantly lower volume relative to the other, posing challenges for separation models to distinguish between them effectively. Thus, to evaluate the performance of our model against established baselines, we manually curated a variant dataset based on the original dataset using the same recordings mixed with different signal to interferences (SIRs). This dataset introduces speaker level imbalances by setting the SIR values to either -5 dB or -10 dB.

6.1.2 WER Results

We have also used the database recorded on the robot’s microphone to verify the algorithm’s applicability to our scenarios. In the network’s offline mode, a significant WER improvements while using NVIDIA RIVA Conformer-based ASR can be deduced from the histogram in Fig. 6.2, analyzing the results for $T_{60} = 0.35 \text{ sec}$ and low-sensor noise scenario. The mean WER was improved from 73.875% (mixed signal) to 18.23% (separated signals). The shift to the left of the histogram after separation further stresses this significant improvement.

Table 6.1 depicts the mean SI-SDR and WER results of this real-world experiments for the proposed algorithm (with and without VAD), and Conv-Tasnet. We discuss here only the results of the offline mode, the online mode will be discussed in the next section.

For the $T_{60} = 0.35 \text{ sec}$ case, it is evident that the proposed algorithm significantly outperforms the Conv-Tasnet and achieves much lower WER. The performance advantages are less pronounced for the $T_{60} = 0.6 \text{ sec}$ case, with our proposed algorithm only slightly outperforming Conv-Tasnet. We also note that Sep-TFAnet performs better than Sep-TFAnet$^{\text{VAD}}$ in both $T_{60}$ conditions.

For the speaker-imbalanced dataset, we present the results in terms of SI-SDR alongside the WER metric for both the weaker and louder speakers. Of particular interest in this experiment is the analysis of the performance of the

\footnote{https://catalog.ngc.nvidia.com/orgs/nvidia/teams/tao/models/speechtotext_en_us_conformer}
This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement No. 871245.

Figure 6.1: Recording setup with the robot ARI at BIU acoustic lab. ARI was positioned at the center of the acoustic lab, with a set of loudspeakers in front of it, on two semi-circles with approximately 1 m and 2 m radius, respectively. In our experiments, we only used the inner semi-circle with five loudspeakers positioned at $[-65, -30, 0, 30, 65]^\circ$. The speech signals were played from two randomly chosen loudspeakers. The lab's computer automatically controlled the entire scenario.

Figure 6.2: Word error rate (WER) improvement with robot data, $T_{60} = 0.35$ s and low sensor noise. The overlap of the speakers was randomly set in the range $[25\%, 50\%]$. Weaker speaker. Our findings indicate that our model consistently performs better than Conv-TasNet in both SIR levels (SIR=-5,-10 dB) and reverberation time settings see Table 6.2.

6.2 Moving Robot: Experiments and Evaluation of the Separation, Noise Reduction, Speaker ID, and Diarisation Modules

6.2.1 Experimental Setup

In this section, we report on a new experimental study to evaluate the audio pipeline. We evaluate the separation track and the noise reduction and diarisation track separately. The CSD and the Arbiter were not tested in this study.

We recorded both static and dynamic scenarios. In the static scenario, ARI was 1 m from two loudspeakers at $[-25, 25]^\circ$, as depicted in Fig. 6.3. In the dynamic scenario, ARI started to move 2 m from the loudspeakers and stopped approximately 1 m from them. The robot was moving in a straight line towards the speakers. We have not yet tested more complex tracks. We note that while maneuvering, the robot may produce stronger ego-noise that can influence the ASR performance. We will examine such scenarios shortly. We either played two speakers with partial overlap (10%-25%) for testing the separation module or non-overlapping speakers for testing the noise reduction and diarisation module. Each sentence was approximately 10 sec long. The speech utterances were drawn from the French set of LibriSpeech database. We also played English utterances for reference.
This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement No. 871245.

Table 6.1: Mean separation results on ARI recorded data.

<table>
<thead>
<tr>
<th>Mode</th>
<th>Algorithm</th>
<th>SI-SDR [dB]</th>
<th>WER [dB]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>↑ 0.35</td>
<td>↓ 0.35</td>
</tr>
<tr>
<td></td>
<td></td>
<td>↑ 0.6</td>
<td>↓ 0.6</td>
</tr>
<tr>
<td>Offline</td>
<td>Unprocessed</td>
<td>-0.01</td>
<td>73.87</td>
</tr>
<tr>
<td></td>
<td>Sep-TFAonet</td>
<td><strong>11.74</strong></td>
<td><strong>18.23</strong></td>
</tr>
<tr>
<td></td>
<td>Sep-TFAonet(^{\text{VAD}})</td>
<td><strong>11.55</strong></td>
<td>18.69</td>
</tr>
<tr>
<td></td>
<td>Conv-Tasnet</td>
<td>10.88</td>
<td>27.6</td>
</tr>
</tbody>
</table>

Table 6.2: Robot data with imbalanced speakers’ volume. The reference for calculating the WER is the reverberated signal.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>↑ 0.35</td>
<td>↓ 0.35</td>
</tr>
<tr>
<td></td>
<td></td>
<td>↑ 0.6</td>
<td>↓ 0.6</td>
</tr>
<tr>
<td>-5</td>
<td>Unprocessed</td>
<td>-4.9/4.9</td>
<td>76.2/75.2</td>
</tr>
<tr>
<td></td>
<td>Sep-TFAonet</td>
<td><strong>10.9/15.6</strong></td>
<td><strong>19.66/10.25</strong></td>
</tr>
<tr>
<td></td>
<td>Conv-Tasnet</td>
<td>9.59/14.62</td>
<td>26.92/23.87</td>
</tr>
<tr>
<td>-10</td>
<td>Unprocessed</td>
<td>-9.92/9.9</td>
<td>80.82/74.5</td>
</tr>
<tr>
<td></td>
<td>Sep-TFAonet</td>
<td><strong>10/18.6</strong></td>
<td><strong>24.87/6.58</strong></td>
</tr>
<tr>
<td></td>
<td>Conv-Tasnet</td>
<td>9.19/18</td>
<td>26.16/16.92</td>
</tr>
</tbody>
</table>

Figure 6.3: ARI 1 meter from speakers.
6.2.2 Audio Separation Track ('Red' System)

In Tables 6.3, 6.4, 6.5, and 6.6 the WER results for the separation modules are depicted. In each table, we depict the results for the mixed and separated signals for five mixture signals, recorded in the lab. For the mixed signal, we list the results with respect to both utterances.

The separation module has two variants. The first is an offline mode, implemented on a PC. In this mode, the entire utterance is processed, resulting in high latency. The second is an online mode implemented on ARI, which is a real-time frame-based solution. The offline mode's performance is usually better.

The static scenarios have a lower WER compared to the dynamic scenarios (Table 6.3 vs. Table 6.4 and Table 6.5 vs. Table 6.6). Notably, the offline separation generally achieved a lower WER compared to the online separation. However, an exception is observed in Table 6.3 for 'Sep (spk2)', where offline performance is inferior to online. Across all our experimental results, English recordings consistently achieved a lower WER compared to the French recordings. Across all results, it is evident that the WER of the mixed audio, when compared to the true speakers’ transcriptions, exhibits the worst-case performance and that the separation module significantly improves the WER performance.

<table>
<thead>
<tr>
<th>Table 6.3: WER results for English data - Static scenario</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Mix (spk1)</strong></td>
</tr>
<tr>
<td>online</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 6.4: WER results for English data - Dynamic scenario</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Mix (spk1)</strong></td>
</tr>
<tr>
<td>online</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 6.5: WER results for French data - Static scenario</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Mix (spk1)</strong></td>
</tr>
<tr>
<td>online</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
</tbody>
</table>

Finally, in Table 6.6, we examplify the transcription of Scenario #5 for both separated speakers as compared to the true transcription, with errors (deletion and insertion) depicted in red.
This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement No. 871245.

### Table 6.6: WER results for French data - Dynamic scenario

<table>
<thead>
<tr>
<th></th>
<th>Mix (spk1)</th>
<th>Mix (spk2)</th>
<th>Sep (spk1)</th>
<th>Sep (spk2)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>online</td>
<td>offline</td>
<td>online</td>
<td>offline</td>
</tr>
<tr>
<td>1</td>
<td>100</td>
<td>63</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>2</td>
<td>100</td>
<td>131</td>
<td>100</td>
<td>58</td>
</tr>
<tr>
<td>3</td>
<td>100</td>
<td>79</td>
<td>100</td>
<td>75</td>
</tr>
<tr>
<td>4</td>
<td>100</td>
<td>117</td>
<td>100</td>
<td>46</td>
</tr>
<tr>
<td>5</td>
<td>100</td>
<td>127</td>
<td>100</td>
<td>55</td>
</tr>
</tbody>
</table>

### Table 6.7: Sample transcription of a sound mixture captured by moving ARI.

<table>
<thead>
<tr>
<th>True Transcription</th>
<th>ASR Transcription</th>
<th>WER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mix qui 100%</td>
<td>pas ma par exemple il est possible que mutine de moi pensait vuillon connaît bien peu son caractère si l'on ne voit pas déjà l'expression sombres et froide que prirent ses regards en répondant à ceux de mathilde</td>
<td>29.72 %</td>
</tr>
<tr>
<td>Speaker #1 il est possible que ce trio se moque de moi pensait julien on connaissait bien peu son caractère si l'on ne voit pas déjà l'expression sombre et froide que prirent ses regards en répondant à ceux de mathilde</td>
<td>pas ma par exemple il est possible que mutine de moi pensait vuillon connaît bien peu son caractère si l'on ne voit pas déjà l'expression sombres et froide que prirent ses regards en répondant à ceux de mathilde</td>
<td>29.72 %</td>
</tr>
<tr>
<td>Speaker #2 hurra my dear vous êtes un homme de parole et d'action vous n'aviez donc pas averti le jury jamais de la vie par exemple bien plus drôle comme ça et pourquoi étiez vous pendant mes salves</td>
<td>comment vous êtes un homme de parole et d'action vous n'aviez donc pas averti julie vous avez de la vie par exemple bien plus drôle comme ça et pourquoi étiez vous pardon les sa</td>
<td>28.94 %</td>
</tr>
</tbody>
</table>

#### 6.2.3 Noise Reduction and Diarisation Track (‘Blue’ System)

Evaluating the ‘blue’ systems involves the transmission of non-overlapping speech utterances. The soundtrack consists of three utterances from the first speaker (‘Speaker A’), followed by three utterances from the second speaker (‘Speaker B’), with a 1 sec pause between them. In this experiment, we used only French utterances and assessed the results for both static (Table 6.8) and dynamic (Table 6.9) scenarios. All WER results, except for the first sentence, are very low, with a clear advantage observed for the static scenario.

Table 6.8: WER results for French data, achieved by the ‘blue’ system in a static scenario.

<table>
<thead>
<tr>
<th>Sentence</th>
<th>Speaker A</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Speaker A</td>
<td>25</td>
<td>0</td>
<td>0</td>
<td>8.3</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Speaker B</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Next, we evaluate the speaker ID module and the induced diarisation performance. The static scenario is demonstrated in Fig. 6.4, and the dynamic scenario in Fig. 6.5. In both cases, six consecutive sentences are shown, with the first three uttered by Speaker A and the last three by Speaker B. This experiment was repeated twice, so each speaker uttered six sentences in total. The results are depicted in separate sub-figures for each scenario (static and dynamic). We color-coded the entire sentence according to the Speaker ID results, providing a clear visual representation of the diarisation performance. In Fig. 6.4 (a), for the static case, the first three utterances of Speaker A were identified properly (dark blue). However, for Speaker B, the Speaker ID module identified an excess speaker. In Fig. 6.4 (b), the Speaker ID module performed well and identified both speakers.

In Fig. 6.5 (a), for the dynamic case, several error types can be observed. For the first three sentences (of Speaker A), there is an excess Speaker ID that is too short to be identified. Hence it is declared as an ‘unknown_voice.’ Then,
Table 6.9: WER results for French data, achieved by the 'blue' system in a dynamic scenario.

<table>
<thead>
<tr>
<th>Sentence</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Speaker A</td>
<td>12.5</td>
<td>0</td>
<td>8.3</td>
<td>8.3</td>
<td>0</td>
<td>22</td>
</tr>
<tr>
<td>Speaker B</td>
<td>100</td>
<td>0</td>
<td>16.3</td>
<td>0</td>
<td>0</td>
<td>16.3</td>
</tr>
</tbody>
</table>

Figure 6.4: Two static experiments.

Figure 6.5: Two dynamic experiments.
the ID of Speaker A's two first utterances was identified properly. However, the third utterance was identified as a new speaker. The last three utterances of Speaker B were identified properly. In Fig. 6.5 (b), the Speaker ID module performed well and identified both speakers correctly.
7 Conclusions

This document describes the audio pipeline in the Spring project. In the audio pipeline, we have two processing tracks: 1) noise reduction and speaker diarisation in a non-overlapping speakers scenario, and 2) speaker separation applied when two speakers are concurrently active. The first track was already extensively analysed in Broca.

We detail on an updated version of the speaker separation module (already discussed in deliverable D3.4) and on the concurrent speaker detector (CSD) which is the core of the ‘Arbiter’ module of the pipeline.

In this document, we report on experiments dynamic scenarios in which ARI is moving towards a group of people (two mannequins in our case, as depicted in Fig. 6.3).

Careful analysis of the results suggests that the two parallel tracks are properly working but that fine-tuning may be required, especially for interactions in the French language. The deployment of the CSD module is expected shortly.
This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement No. 871245.

Bibliography


