香港中文大學（深圳）
The Chinese University of Hong Kong, Shenzhen

Representation Learning and Cognitive Modeling of Speech in Real World Scenarios

Junyi Ao
敖君逸

A Thesis Submitted in Partial Fulfilment
of the Requirements for the Degree of
Doctor of Philosophy
in
Computer Science

The Chinese University of Hong Kong, Shenzhen

April 2026

Thesis Assessment Committee

Professor Satoshi Nakamura (Chair)

Professor Haizhou Li (Thesis Supervisor)

Professor Zhizheng Wu (Committee Member)

Professor Wanli Ouyang (Examiner from CUHK)

Professor Eng-Siong Chng (External Examiner)

Abstract

of thesis entitled:

Representation Learning and Cognitive Modeling of Speech in Real World Scenarios

Submitted by Junyi Ao

for the degree of Doctor of Philosophy

at The Chinese University of Hong Kong, Shenzhen in April 2026

Speech intelligence has evolved from traditional supervised speech modeling toward a broader paradigm that emphasizes general-purpose representation learning, robustness in realistic acoustic conditions, and higher-level spoken interaction. In this context, this dissertation studies *representation learning and cognitive modeling of speech in real-world scenarios*. Rather than focusing on a single end-to-end architecture, it investigates speech intelligence across multiple levels of abstraction, progressing from foundational speech representation learning, to robust modeling in complex acoustic environments, and finally to spoken language systems capable of perceiving and reasoning beyond lexical content.

The first part of the dissertation studies speech-only self-supervised pre-training with discrete units. It investigates how discrete speech codes can bridge raw continuous speech and higher-level pre-training objectives when paired text is unavailable. In particular, CoBERT learns speech representations by predicting contextualized code representations from masked speech, while Speech2C extends speech-only

pre-training to the decoder side by using pseudo codes as reconstruction targets for encoder-decoder models. Together, these studies show how discrete units can support both encoder representation learning and decoder pre-training under speech-only conditions.

The second part moves from generic representation learning to robust speech modeling in realistic multi-speaker scenarios. It covers speaker-aware self-supervised learning for mixture speech and a unified framework for speaker extraction and diarization. This part emphasizes that effective speech intelligence requires not only informative representations, but also robustness to overlap, interference, and speaker ambiguity in practical acoustic environments.

The third part shifts the focus from robust signal processing to cognitive spoken perception and interaction. To move beyond transcript-based understanding, the dissertation introduces SD-Eval, a benchmark for spoken dialogue understanding beyond words, and presents SOLLA, a speech-oriented large language model that jointly models spoken instructions and acoustic context. Building on this direction, the dissertation further studies low-latency spoken reasoning through a think-aloud framework in which reasoning and speaking proceed asynchronously, improving responsiveness while maintaining strong reasoning ability.

Taken together, this dissertation presents a unified view of speech intelligence that progresses from representation learning, to robust speech modeling in realistic scenarios, and ultimately to cognitively capable spoken language systems with beyond-word perception and low-latency reasoning abilities. It contributes both foundational modeling methods and higher-level system designs for more natural, robust, and intelligent speech-based human-computer interaction.

摘要

面向真实世界场景的语音表征学习与认知建模

语音智能研究正从传统的监督式语音建模，逐步走向更强调通用表征学习、真实场景鲁棒性以及高层次口语交互能力的研究范式。本文以“面向真实世界场景的语音表征学习与认知建模”为主题，从多个抽象层次研究语音智能：首先关注基础语音表征学习，其次关注复杂声学环境中的鲁棒建模，最后进一步探索具备超越文字感知与推理能力的口语语言系统。

论文第一部分研究基于离散单元的纯语音自监督预训练，关注在缺乏配对文本监督的条件下，如何利用离散语音单元连接底层连续语音信号与高层预训练目标。具体而言，本文提出 CoBERT，通过从被遮蔽语音中预测上下文化的编码表征来学习更强的语音表示；同时提出 Speech2C，将纯语音预训练扩展到解码器端，利用离散语音单元序列对编码器—解码器模型进行预训练。这一部分说明了离散语音单元如何同时支持编码器表征学习与解码器预训练。

论文第二部分进一步研究真实多说话人场景中的鲁棒语音建模，包括面向混叠语音的说话人感知自监督预训练，以及统一的说话人提取与说话人日记化框架。该部分表明，语音智能不仅需要有效表征，还需要在重叠、干扰和说话人不确定性等真实声学条件下保持鲁棒性。

论文第三部分将重点从鲁棒信号处理推进到口语语言模型的认知感知与交互能力。为突破仅依赖转写文本的理解方式，本文构建了 SD-Eval 基准，用于评测语音对话中“超越文字”的理解能力，并提出 Solla 模型，对语音指令与声学上

下文进行联合建模。在此基础上，本文进一步研究低延迟口语推理，提出边想边说框架，使推理与发声能够异步并行，从而在保持较强推理能力的同时提升语音交互的响应性。

总体而言，本文构建了一条从语音表征学习，到真实场景鲁棒建模，再到具备超越文字感知与低延迟推理能力的认知型口语语言系统的研究脉络。论文既提出了基础建模方法，也探索了更高层次的系统设计，为构建更自然、更鲁棒、更智能的语音交互系统提供了系统性的研究视角。

Acknowledgement

I would like to express my deepest gratitude to my supervisor, Professor Haizhou Li, for his continuous guidance, encouragement, and support throughout my Ph.D. study. His rigorous attitude toward research, broad vision, and constant encouragement have profoundly influenced my academic growth. I am sincerely thankful for the freedom he gave me to explore different research directions, as well as for his valuable advice at every important stage of my doctoral journey.

I am also very grateful to the professors and collaborators who have worked with me on various research projects, including Professor Zhizheng Wu, Professor Tom Ko, and Professor Shuai Wang. Their insightful discussions, generous support, and constructive feedback have greatly shaped this dissertation and broadened my understanding of the field. I have learned a great deal from each collaboration.

My sincere thanks also go to my mentors and leaders during my internships, including Dr. Shujie Liu, Dr. Long Zhou, Dr. Xiaohai Tian, Mr. Kainan Peng, Dr. Mingbo Ma, and Dr. Qing He. I was fortunate to have the opportunity to work with them in inspiring research environments. Their guidance, trust, and high standards not only helped me improve as a researcher, but also gave me valuable perspectives on how research can create impact in real-world systems.

I would also like to thank my labmates and friends in the laboratory for their companionship and support over the years. I am grateful for the many discussions, collaborations, and moments of encouragement we shared. The friendly and stimulat-

ing environment in the lab made my Ph.D. life much more enjoyable and meaningful.

Finally, and most importantly, I would like to thank my family for their unconditional love, understanding, and support. They have always been my strongest source of strength, giving me confidence and courage throughout this long journey. I am especially grateful to my girlfriend for her love, patience, and constant encouragement. Her support has been invaluable during both the joyful and difficult moments of my Ph.D. life.

To all of you, I offer my heartfelt thanks.

Contents

List of Figures

List of Tables

Acronyms

ASR	automatic speech recognition
SSL	self-supervised learning
LLM	large language model
SLM	spoken language model
TTS	text-to-speech
CTC	connectionist temporal classification
LM	language model
WER	word error rate
cpWER	concatenated minimum-permutation word error rate
DER	diarization error rate
PIT	permutation invariant training
BCE	binary cross-entropy
CE	cross-entropy
PESQ	perceptual evaluation of speech quality
STOI	short-time objective intelligibility
SI-SDR	scale-invariant signal-to-distortion ratio
SI-SDRi	scale-invariant signal-to-distortion ratio improvement
LoRA	low-rank adaptation

CoT	chain-of-thought
SFT	supervised fine-tuning
DPO	direct preference optimization
VAD	voice activity detection
EEND	end-to-end neural diarization
TS-VAD	target-speaker voice activity detection
TCN	temporal convolutional network
CNN	convolutional neural network
RNN-T	recurrent neural network transducer
AQA	audio question answering
S2T	speech-to-text
S2S	speech-to-speech

Chapter 1

Introduction

Summary

This chapter motivates the thesis topic of representation learning and cognitive modeling of speech in real-world scenarios. It reviews relevant background and related work, summarizes the main contributions, and sketches the organization across Chapters 2–6.

1.1 Background and Motivation

Speech has long been one of the most natural and fundamental forms of human communication. Building intelligent systems that can process and interact through speech has therefore been a central goal in speech and language research. Over the past decades, this area has undergone a substantial evolution. Early studies mainly focused on supervised speech modeling. While such systems achieved remarkable progress, they were largely designed around narrow tasks and often depended on substantial amounts of labeled data.

More recently, the emergence of self-supervised learning has significantly changed the landscape of speech research. Instead of relying solely on manual annotations, self-supervised methods learn general-purpose representations directly from large-scale unlabeled speech corpora [Schneider et al., 2019; Baevski et al., 2020b; Hsu et al., 2021; Chen et al., 2022a; Baevski et al., 2022]. This shift has enabled speech models to capture richer acoustic and phonetic patterns, and has led to strong transferability across downstream tasks. In this context, a central research question becomes how to learn effective speech representations from raw speech signals, especially under limited or even absent text supervision. This challenge motivates the first part of this dissertation, which studies speech-only self-supervised pre-training and the role of discrete units in bridging low-level acoustic signals and higher-level representational learning.

However, learning strong representations alone does not suffice for practical speech systems. Real-world speech rarely occurs in clean and isolated conditions. Instead, speech is often mixed with overlapping speakers, environmental noise, and other forms of acoustic interference. As speech technologies move from controlled benchmarks toward practical deployment, robustness under realistic acoustic conditions becomes increasingly important [Chen et al., 2022a; Medennikov et al., 2020; Maiti et al., 2023; Pan et al., 2022]. This raises the next key question: how can speech models remain effective when the input speech is corrupted by overlap, interference, or speaker ambiguity? Addressing this challenge requires not only better representation learning, but also speech modeling methods that can operate reliably in complex real-world scenarios. This motivates the second part of this dissertation, which investigates robust speech modeling in multi-speaker acoustic environments.

At the same time, the goals of spoken systems are also expanding. Traditional speech systems have often treated speech primarily as an input modality for transcription or semantic parsing. Yet human spoken communication contains much

richer information than lexical content alone. Paralinguistic cues, speaker traits, emotion, speaking style, and surrounding acoustic context all contribute to human understanding in spoken interaction [Busso et al., 2008; McKeown et al., 2010; Poria et al., 2019; Xue et al., 2023; Lin et al., 2024]. Moreover, with the rapid development of large language models and speech-oriented large language models, spoken systems are increasingly expected not only to recognize or transcribe speech, but also to perceive contextual information beyond words and to support more natural, interactive, and cognitively capable spoken communication [Chu et al., 2024; Tang et al., 2024; Hu et al., 2024; Défossez et al., 2024; Fang et al., 2024; Xie and Wu, 2024; Xu et al., 2025].

This trend motivates a further shift from robust speech processing toward cognitive modeling for spoken language systems. In this dissertation, the term "cognitive modeling" does not refer to modeling human cognition directly, but rather to endowing spoken language systems with higher-level capabilities such as beyond-word perception, contextual interpretation, and low-latency reasoning in spoken interaction. From this perspective, the studies in this dissertation span multiple levels of speech intelligence: from foundational representation learning, to robust modeling in realistic acoustic conditions, and finally to spoken language systems capable of richer perception and reasoning.

Although the chapters in this dissertation do not all target the same end-to-end architecture, they address a connected research agenda. The earlier chapters focus on foundational problems in speech representation and robustness, which determine how speech signals can be effectively encoded and processed under realistic conditions. The later chapters shift to spoken language models, where speech is treated not only as an acoustic signal to be recognized, but also as a rich modality that conveys information beyond text and supports natural spoken interaction. Together, these studies reflect a progression from signal-level representation, to real-world robust-

ness, and ultimately to cognitive spoken interaction.

1.2 Scope and Research Questions

Based on the above motivation, this dissertation studies speech intelligence from two tightly connected perspectives. The first perspective concerns foundational speech representation learning and robust speech modeling under realistic conditions. The second concerns higher-level cognitive capabilities in spoken language systems, especially the ability to perceive and reason beyond lexical content.

More specifically, this dissertation is organized around the following research questions:

- How can effective speech representations be learned from speech-only data, especially when no paired text supervision is available?
- How can discrete speech units support self-supervised pre-training for both encoder representation learning and decoder pre-training?
- How can speech modeling methods remain robust in complex real-world scenarios such as overlapping multi-speaker speech?
- How can spoken language models perceive information beyond lexical transcripts, including paralinguistic cues and acoustic context?
- How can spoken language models support more natural and low-latency spoken reasoning and interaction?

These questions together define the scope of this dissertation. Rather than focusing on a single model family, this dissertation studies how speech systems evolve

from foundational representation learning to robust real-world modeling, and further toward spoken language systems with perceptual and cognitive abilities beyond words.

1.3 Dissertation Overview

The dissertation is organized into four technical chapters following a progression from foundational modeling to higher-level spoken intelligence.

Chapter 2 studies speech-only self-supervised pre-training with discrete units. It begins with representation learning through discrete code prediction, and then extends this idea to decoder pre-training using pseudo codes without text supervision. This chapter addresses the foundational problem of how to learn useful speech representations and pre-training objectives when only unlabeled speech is available.

Chapter 3 moves from generic speech representation learning to robust speech modeling in realistic multi-speaker environments. It first discusses speaker-aware self-supervised learning for mixture speech, and then presents a unified framework for speaker extraction and diarization. This chapter emphasizes that effective speech intelligence requires not only good representations, but also robustness to overlap and interference in real-world acoustic conditions.

Chapter 4 shifts the focus from robust speech modeling to beyond-word perception in spoken language models. It first introduces a benchmark for evaluating spoken dialogue understanding beyond lexical content, and then presents a speech-oriented large language model that explicitly incorporates acoustic context. This chapter marks the transition from representation and robustness toward higher-level cognitive spoken perception.

Chapter 5 further extends this direction by studying low-latency reasoning in spoken language models. It investigates the limitations of conventional think-then-speak

interaction and introduces a framework in which reasoning and speaking proceed asynchronously. This chapter represents the final stage of the dissertation, where spoken systems are expected not only to perceive richer information, but also to reason and interact naturally in speech form.

Finally, Chapter 6 concludes the dissertation and discusses future directions, including real-time spoken reasoning, more unified speech–language systems, and broader forms of multimodal spoken intelligence.

Overall, the dissertation follows a coherent progression:

$$\textit{speech representation learning} \rightarrow \textit{robust speech modeling} \rightarrow \textit{beyond-word perception} \rightarrow \textit{spoken reasoning and interaction.}$$

1.4 Summary of Contributions

The main contributions of this dissertation can be summarized as follows.

1. **Speech-only self-supervised pre-training with discrete units.** This dissertation investigates how discrete speech units can support self-supervised learning without paired text supervision. It studies both encoder-side representation learning through discrete code prediction and decoder-side pre-training through pseudo-code reconstruction, providing a unified perspective on speech-only pre-training with discrete units.

2. **Robust speech modeling in multi-speaker real-world scenarios.** This dissertation extends speech modeling from generic pre-training settings to acoustically challenging multi-speaker scenarios. It includes speaker-aware self-supervised learning for mixture speech and a unified framework for speaker extraction and diarization, contributing to more robust speech processing under realistic overlap conditions.

3. **Benchmarking and modeling perception beyond words in spoken language models.** This dissertation studies how spoken language models can move beyond transcript-based understanding. It contributes both an evaluation benchmark for spoken dialogue understanding beyond lexical content and a model architecture that incorporates acoustic context into speech-oriented large language modeling.

4. **Low-latency spoken reasoning through think-aloud modeling.** This dissertation proposes a framework for reducing the latency of spoken reasoning by allowing reasoning and speech generation to proceed asynchronously. This contributes to more natural spoken interaction and offers a system perspective on how reasoning ability can be exposed and coordinated in real time.

Taken together, these contributions present a unified view of speech intelligence across multiple levels of abstraction, from foundational representation learning to cognitively capable spoken language systems.

1.5 Publications Underlying This Dissertation

The technical chapters of this dissertation are based on the following publications and manuscripts. This list includes only works that directly underlie the main body of the dissertation; a complete publication list during my Ph.D. study is provided in Appendix A.

Chapter 2: Speech-Only Self-Supervised Pre-training with Discrete Units

- Meng, C.*, **Ao, J***., Ko, T., Wang, M., and Li, H. (2023). CoBERT: Self-Supervised Speech Representation Learning Through Code Representation Learning. In

Interspeech 2023, pages 2978–2982. * Equal contribution.

- **Ao, J.**, Zhang, Z., Zhou, L., Liu, S., Li, H., Ko, T., Dai, L., Li, J., Qian, Y., and Wei, F. (2022b). Pre-Training Transformer Decoder for End-to-End ASR Model with Unpaired Speech Data. In *Interspeech 2022*, pages 2658–2662.

Chapter 3: Robust Speech Modeling in Multi-Speaker Real-World Scenarios

- Lin, J., Ge, M., **Ao, J.**, Deng, L., and Li, H. (2024b). SA-WavLM: Speaker-Aware Self-Supervised Pre-Training for Mixture Speech. In *Interspeech 2024*, pages 597–601.

- **Ao, J.**, Yıldırım, M. S., Tao, R., Ge, M., Wang, S., Qian, Y., and Li, H. (2025). USED: Universal Speaker Extraction and Diarization. *IEEE Transactions on Audio, Speech and Language Processing*, 33:96–110.

Chapter 4: Perception Beyond Words in Spoken Language Models

- **Ao, J.**, Wang, Y., Tian, X., Chen, D., Zhang, J., Lu, L., Wang, Y., Li, H., and Wu, Z. (2024). SD-Eval: A Benchmark Dataset for Spoken Dialogue Understanding Beyond Words. In *The Thirty-eighth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

- **Ao, J.**, Chen, D., Tian, X., Feng, W., Zhang, J., Lu, L., Wang, Y., Li, H., and Wu, Z. (2026a). Solla: Towards a Speech-Oriented LLM That Hears Acoustic Context. Under review for *Interspeech 2026*.

Chapter 5: Spoken Language Models that Think Aloud

- **Ao, J.**, Peng, K., Liu, X., Zhang, S., Tang, Z., Ma, X., Li, X., Wang, Y., Wu, Z., Li, H., He, Q., and Ma, M. (2026b). Spoken Language Models that Think Aloud. Under review for *Interspeech 2026*.

Summary

This chapter motivates the thesis topic of representation learning and cognitive modeling of speech in real-world scenarios. It reviews relevant background and related work, summarizes the main contributions, and sketches the organization across Chapters 2–6.

□ End of chapter.

Chapter 2

Speech-Only Self-Supervised Pre-training with Discrete Units

Summary

This chapter presents two lines of work on learning from discrete speech codes. The first section introduces CoBERT, which learns speech representations by predicting code representations from masked speech. The second section presents Speech2C, which pre-trains an encoder–decoder for speech recognition using pseudo codes from clustering, without using text to pre-train the decoder.

This chapter begins the technical part of the dissertation by addressing the most foundational question in speech intelligence: how to learn effective speech representations from unlabeled speech signals. As discussed in Chapter 1, recent progress in speech research has been driven by self-supervised learning, which aims to reduce the dependence on large-scale manual annotation while improving the generality

and transferability of learned representations. In this context, this chapter focuses on speech-only self-supervised pre-training with discrete units, investigating how discrete speech codes can serve as an intermediate bridge between raw continuous speech and higher-level pre-training objectives.

More specifically, this chapter studies two complementary directions. The first explores how discrete code representations can improve encoder-side speech representation learning, while the second extends the use of discrete units to decoder-side pre-training in encoder–decoder ASR models without relying on paired text. Together, these studies establish the dissertation's starting point: before speech systems can become robust or cognitively capable, they must first learn informative and transferable representations from speech itself.

2.1 Self-Supervised Speech Representation Learning through Discrete Codes

2.1.1 Introduction

Following the success of BERT in natural language processing [Devlin et al., 2019], self-supervised speech representation learning has emerged as a major direction in speech research. Its central premise is that stronger learned speech representations can simplify downstream modeling and thereby reduce the amount of annotated data required for supervised fine-tuning. Under this motivation, a broad range of training strategies has been developed [Oord et al., 2018; Schneider et al., 2019; Chung and Glass, 2020; Hsu et al., 2021; Chen et al., 2022a; Baevski et al., 2022; Cheng et al., 2022; Ma et al., 2023a], including wav2vec [Baevski et al., 2020a; Baevski and Mohamed, 2020; Baevski et al., 2020b], HuBERT [Hsu et al., 2021], and more recently data2vec [Baevski et al., 2022].

Beyond the use of unpaired speech in self-supervised learning, prior studies have also shown that unpaired text data [Ao et al., 2022a; Bapna et al., 2021; Tang et al., 2022] can provide useful supervision. In this line of work, speech-text pre-training has produced promising results by seeking unified representations through joint training between speech and text. From a broader perspective, these methods can be understood as attempts to narrow the representational gap between the two modalities. A common difficulty, however, is the alignment of unpaired speech and text representations, and this challenge can be alleviated when paired speech recognition data are available [Ye et al., 2022].

Against this background, this work proposes the **Co**de **BERT** (CoBERT) approach for self-supervised speech representation learning. The method is motivated by both recent progress in speech-text pre-training and the data2vec framework. First, we posit that the gains from speech-text pre-training arise from improvements in the speech encoder, which captures additional information from text representations. Although complete transcripts for unpaired speech data are difficult to obtain, recent studies [Ao et al., 2022b] indicate that self-organized codes are closely related to the text of the original speech. Second, data2vec promotes the prediction of contextualized latent representations within a self-distillation framework. Based on these observations, we investigate code representation learning as a means to improve speech representation learning. The central idea is to convert speech into a sequence of discrete units (codes) and to learn code representations by predicting them from a masked view of the original speech input. Because these codes are derived directly from unpaired speech data, the alignment issue that arises in speech-text pre-training can be avoided.

The contributions of this work are threefold. First, we explore code representation learning in a self-distillation framework to improve speech representation learning. Second, CoBERT is able to predict contextualized latent representations from different

and multiple modalities simultaneously. Third, experiments show that, relative to data2vec, CoBERT reduces the word error rate (WER) by 6.7% on the LibriSpeech subsets [Panayotov et al., 2015a] and yields substantial gains on the SUPERB ST task [Tsai et al., 2022], with an improvement of around 1.35 BLEU.

2.1.2 Related Work

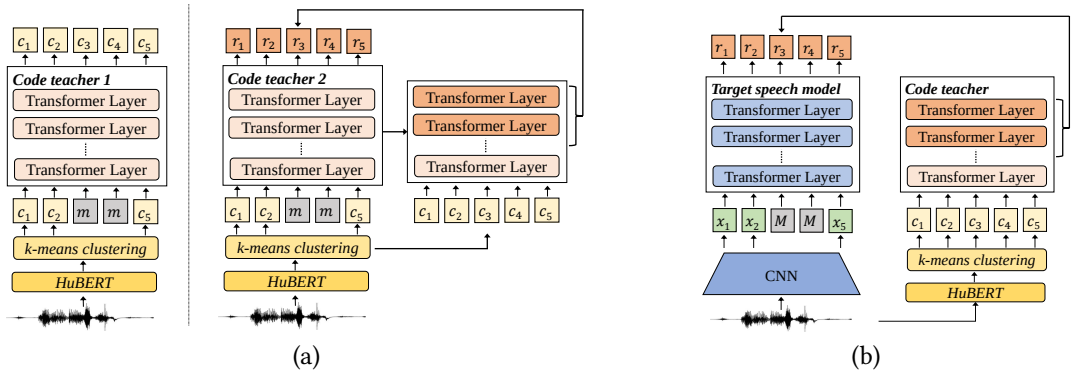
CoBERT is motivated by and most related to vq-wav2vec [Baevski et al., 2020a], HuBERT [Hsu et al., 2021] and data2vec [Baevski et al., 2022]. vq-wav2vec is a pioneer work in exploring code representation learning. Further, they show that BERT-based representation learning is more effective in discrete than in continuous signal spaces [Baevski and Mohamed, 2020]. HuBERT proposes to generate codes with an offline clustering step. Then it performs a BERT-based pre-training using these codes as target labels. data2vec outperforms BERT-based pre-training methods by promoting a direct prediction of contextualized latent representations instead of modality-specific targets. Our method integrates techniques of these methods.

We follow HuBERT's clustering procedures to obtain the codes so that CoBERT can perform representation learning completely in discrete code space. In comparison to data2vec, we follow their distillation approach to transfer the knowledge learnt from the code to the speech encoder, but our method differs in that CoBERT predicts representations of a different modality. The teacher is a code encoder but the student is a speech encoder. Moreover, CoBERT obtains better results when it predicts two sets of representations, one from the speech and one from the code, at the same time.

Another line of research aims at utilizing unpaired text to benefit speech representation learning [Ao et al., 2022a; Bapna et al., 2021; Tang et al., 2022]. In this work, we do not compare our results with these methods as we pre-train models only with unpaired speech.

2.1.3 Method

Our approach (Fig. 2.1) can be divided into three steps: code generation, code representation learning and knowledge distillation.



(a) (b)

Figure 2.1: Illustration of CoBERT. The input codes for the code teachers are obtained by k-means clustering on HuBERT features. (a) We examine two code representation learning methods. The left one is based on BERT and the right one is based on data2vec. (b) The representations produced by the code teacher are regressed by the target speech encoder given a masked view of the speech input.

2.1.3.1 Code Generation

We follow the hidden units discovery steps described in [Hsu et al., 2021] to extract the codes. More precisely, we feed the raw waveform into a HuBERT model, and then apply k-means clustering to the latent features at a certain transformer layer. The generated codes are denoted as $C = [c_1, ..., c_T]$ where T is the number of frames, $c_t \in [K]$, and K is the number of k-means classes.

We obtain codes from the 6$^{\text{th}}$ layer features of HuBERT BASE-it1 as well as 9$^{\text{th}}$ layer features of HuBERT BASE-it2. We train HuBERT BASE-it1 ourselves, with the raw waveform as inputs and k-means clustering results on Mel-frequency cepstral coefficients (MFCC) features as targets. We use the BASE model released by Fairseq[1]

[1] https://dl.fbaipublicfiles.com/hubert/hubert_base_ls960.pt

as the HuBERT BASE-it2 model. As shown in Table 2.2, codes from HuBERT BASE it2-L9 have better quality, so we use them in the subsequent steps.

2.1.3.2 Code Representation Learning

We examine code representation learning in two different ways (Fig. 2.1(a)).

Code teacher 1 is based on mask language modeling (MLM) similar to BERT [Devlin et al., 2019]. Let Z denote the indices where the elements are to be masked. Also let $\widetilde{C} = mask(C, Z)$, which replaces the codes in C whose indices lie in Z with a special code M. Then the model is trained to predict the masked codes. We only compute the loss over the masked positions, which is

$$L_{mlm} = -\sum_{t \in Z} \log p(c_t | \widetilde{C}, t). \tag{2.1}$$

We use the same HuBERT BASE model as in [Hsu et al., 2021], except that we replace the waveform encoder with a word embedding layer.

Code teacher 2 adopts the self-distillation approach of data2vec [Baevski et al., 2022]. The teacher model takes C as input and produces target representations $R = [r_1, ..., r_T] = \frac{1}{L} \sum_{l=N-L+1}^{N} \hat{a}^l$, where N is the number of transformer layers and features from each of the top L layers will be normalized to $\hat{a}^l$ then summed to targets. The student model takes in $\widetilde{C}$ and regresses R.

The parameters of the teacher model are exponentially moving average (EMA) of those of the student model: $\theta_t \leftarrow \tau\theta_t + (1-\tau)\theta_s$, where θ_t is the parameters of the teacher model and θ_s is the parameters of the student model. Only the student model is updated via backpropagation. The objective is an $L2$ loss,

$$L_{sd} = \frac{1}{2}(\hat{R} - R)^2. \tag{2.2}$$

where $\hat{R}$ is the predictions of the student model and sd denotes self-distillation.

For code teacher 2, we apply the data2vec BASE model with its waveform encoder replaced with a word embedding layer. We use the features from the top 8 layers to generate targets, i.e., $L = 8$. We adopt slightly different masking strategies for the two teachers. Let p denote the probability of an input element chosen as the start position of a mask span. We adopt $p = 8\%$ for MLM based training and $p = 6.5\%$ for self-distillation based training. For both models, we set their mask spans to 10.

2.1.3.3 Knowledge Distillation

The target speech encoder, CoBERT, is trained by predicting the representations of the code input given a masked view of the speech input (Fig. 2.1(b)). In other words, we distil the code model into a speech model.

In this work, the architecture of the CoBERT model follows exactly the speech data2vec BASE model [Baevski et al., 2022]. It takes raw waveform as input. Let $X = [x_1, \ldots, x_T]$ denote the speech hidden states after the waveform is downsampled by the same waveform encoder as the one described in [Baevski et al., 2022]. We mask X with $p = 6.5\%$ and mask span 10 to generate a masked view $\widetilde{X} = mask(X, Z)$. The pre-trained code teacher produces code representations R_{code} based on the aligned code sequence corresponding to X. The objective is to regress the code representations:

$$L_{code} = \frac{1}{2}(R_{Co} - R_{code})^2 \tag{2.3}$$

where R_{Co} is the representations generated by the CoBERT model.

Additionally, to fully utilize the speech information, our distillation process can be combined with the self-distillation setup used in data2vec. The teacher model, which averages the parameters of the CoBERT model exponentially, takes X as input and produces speech representations R_{speech} from its top 8 layers. We use a separate linear layer on top of CoBERT model, which generates $R_{Co'}$, to regress R_{speech}. The

loss from the self-distillation teacher is therefore

$$L_{speech} = \frac{1}{2}(R_{Co'} - R_{speech})^2. \tag{2.4}$$

The final loss for CoBERT is a weighted sum of L_{code} and L_{speech}:

$$L_{CoBERT} = \alpha L_{code} + (1 - \alpha) L_{speech}. \tag{2.5}$$

where $0 \leq \alpha \leq 1$. In all of our experiments, we set $\alpha = 0.5$.

2.1.4 Experiments

2.1.4.1 Experiment Setup

We conduct our experiments using Fairseq[2] [Ott et al., 2019]. For pre-training, we mainly follow the training process and hyper-parameters in [Hsu et al., 2021; Baevski et al., 2022]. We use the full 960 hours training set of LibriSpeech [Panayotov et al., 2015a] without the transcription for pre-training. For code teacher 1, we optimize the model with Adam [Kingma and Ba, 2014] by warming up the learning rate for the first 8% of updates to a peak of 5×10^{-4}, which is linearly decayed for the following updates. For code teacher 2 and CoBERT experiments, we use Adam [Kingma and Ba, 2014] to optimize the model with a tri-stage scheduler, which linearly warms up the learning rate for the first 3% of updates to a peak of 5×10^{-4}, holds it for 90% of updates and linearly decays the learning rate for the following updates. We pre-train the model for 400k updates on 16 GPUs with a batch size of around 10k tokens, 11.85k tokens and 237s samples per GPU for code teacher 1, code teacher 2 and CoBERT experiments.

For the ASR fine-tuning, we utilize 10 and 100 hours subsets, and segment the text

[2] https://github.com/pytorch/fairseq

with the character set. The learning rate is warmed up for the first 10% steps, held as a constant for the following 40% steps, and decayed linearly for the rest steps. For the 10/100 hours subset, we train the model with a learning rate of 5e-5/3e-5 for 20k/80k. For the 10 hours subset, the encoder part is fixed for the first 10k steps.

We evaluate our model on the ASR task by using wav2letter++ [Pratap et al., 2019] beam search decoder with a beam size of 1500 for 4-gram language model-fused decoding, which optimized:

$$\log P_{ctc}(Y|X) + \omega_1 \log P_{LM}(Y) + \omega_2|Y| \tag{2.6}$$

where Y is the prediction of a text sequence, $|Y|$ is the sequence length, and ω_1 and ω_2 are the weights of the language model and word score, respectively. The decoding hyperparameters, including language model weight, word score and silence weight, are searched with Ax[3], which is a Bayesian optimization toolkit. We use the official 4-gram language model[4] trained on the LibriSpeech-LM corpus for inference.

For the SUPERB ASR and ST fine-tuning, the training and inference processes follow [Tsai et al., 2022]. We use LibriSpeech and CoVoST2 en→de dataset [Wang et al., 2020a] for evaluation. The downstream model for ST has three transformer layers for both the encoder and decoder, with an additional convolution layer for down-sampling the input. The downstream model for ASR has two layers of BLSTM and is optimized by CTC loss on characters level.

2.1.4.2 Main Results

Table 2.1 shows the main results of code teachers and CoBERT on ASR and ST tasks. We evaluate the WER scores on the standard LibriSpeech test-clean and test-other sets and BLEU scores on the test set of the CoVoST2 En → De dataset. We compare

[3] https://github.com/facebook/Ax
[4] https://www.openslr.org/resources/11/4-gram.arpa.gz

Table 2.1: Main results on ASR and ST tasks. The WER scores are evaluated on the test-clean and test-other sets of LibriSpeech when using the 10 hours or 100 hours subsets as the training data and the BLEU scores are evaluated on the test set of the CoVoST2 En → De dataset. The WER scores of HuBERT and data2vec without LM and the BLEU score of data2vec are obtained by fine-tuning the released model as they are not reported in the corresponding papers.

Model	10 hours subset				100 hours subset				SUPERB ASR	SUPERB ST
	No LM		4-gram		No LM		4-gram			
	clean	other	clean	other	clean	other	clean	other		
Baselines										
DiscreteBERT [Baevski and Mohamed, 2020]	-	-	5.9	14.1	-	-	4.5	12.1	-	-
wav2vec 2.0 Base [Baevski et al., 2020b]	11.1	17.6	4.3	9.5	6.1	13.3	3.4	8.0	6.43	14.81
HuBERT Base [Hsu et al., 2021]	10.1	16.8	4.3	9.4	5.6	12.7	3.4	8.1	6.42	15.53
WavLM Base [Chen et al., 2022a]	9.8	16.0	4.3	9.2	5.7	12.0	3.4	7.7	6.21	-
data2vec Base [Baevski et al., 2022]	7.2	12.3	3.9	8.1	4.2	9.7	**2.8**	6.8	4.94	17.56
Our Methods										
code teacher 1 (BERT)	8.8	14.5	5.1	9.5	5.0	10.8	3.6	8.1	-	17.11
code teacher 2 (data2vec)	7.9	12.9	4.4	8.8	4.6	9.9	3.4	7.5	-	17.75
CoBERT										
- distilling code teacher 1	7.4	12.7	3.7	7.9	4.3	9.8	2.9	6.8	-	17.81
+ self-distillation	7.2	12.2	3.7	7.7	4.3	9.6	3.0	6.6	-	18.12
- distilling code teacher 2	7.5	12.3	4.0	8.0	4.2	9.3	3.0	6.7	-	18.73
+ self-distillation	**6.8**	**11.4**	**3.6**	**7.4**	**4.0**	**8.9**	**2.8**	**6.4**	**4.74**	**19.10**

our method with several works from the literature, including DiscreteBERT [Baevski and Mohamed, 2020], wav2vec 2.0 [Baevski et al., 2020b], HuBERT [Hsu et al., 2021], WavLM [Chen et al., 2022a] and data2vec [Baevski et al., 2022], which are competitive self-supervised approaches. As the WER scores without LM of HuBERT and data2vec and the BLEU scores of data2vec are not reported in the corresponding papers, the results are obtained by fine-tuning the released model.

Experiments show that without LM fusion, code teacher 1 outperforms HuBERT on the ASR task, which relatively brings 12.1% and 14.2% reductions on the test-clean and test-other sets, while it performs worse or on par compared to the HuBERT baseline with LM fusion. For the ST task, code teacher 1 achieves significant improvement by around 1.58 BLEU. This indicates the code representations of code teacher 1 may carry extra semantic information. In consideration with a similar observation in [Baevski and Mohamed, 2020], we hypothesize that BERT-based representation learning may be more effective in discrete than in continuous signal spaces.

Without self-distillation, CoBERT models outperform DiscreteBERT, HuBERT and wav2vec 2.0 by a large margin for both the ASR and ST tasks, while performing better than data2vec on the ST task. With self-distillation, CoBERT reaches a relatively 6.7% WER reduction on the averages of all ASR sets and improves the BLEU score from 17.56 to 19.10 compared to the data2vec baseline.

2.1.4.3 Analysis

Code quality across different features. As code quality may decide the final performance, we investigate the clustering quality across different features, including features of the 6th layer of HuBERT BASE-it1, the 9th layer of HuBERT BASE-it2, the 9th layer of data2vec, and the average top 8 layers of data2vec. We generate frame-level phonetic transcripts using Montreal Forced Aligner [McAuliffe et al., 2017] with the default pre-trained English acoustic model [McAuliffe and Sonderegger, 2022] and dictionary [Gorman et al., 2011]. Then we follow [Hsu et al., 2021] to compute the code quality metrics using the codes and the phonemes. The results are shown in Table 2.2. Although the codes from it2-L9 are better than those from it1-L6, they achieve similar performance on the test-other subset of LibriSpeech. Therefore, it is fair for us to compare the performance of CoBERT and HuBERT. We also explore whether we can derive better-quality codes from the data2vec model since it has a better ASR performance. However, it turns out that the qualities of codes from data2vec are much worse than HuBERT's.

Effect of different teachers and model input. In this section, we verify the importance of the code teachers by using different teachers for the distillation. In Table 2.3, the parameters of the HuBERT model and code teachers are fixed throughout the distillation. We intentionally not to further distil the data2vec checkpoint in Table 2.1 as it is already trained with self-distillation.

Table 2.2: Quality of the clustering assignments and the effect on the ASR task with respect to different features.

model	feature	phone purity	cluster purity	PNMI	WER
HuBERT	it1-L6	0.668	0.110	0.657	12.74
	it2-L9	0.674	0.107	0.666	12.73
data2vec	it1-L9	0.583	0.100	0.594	-
	Top8-avg	0.536	0.111	0.540	-

Firstly, we want to show that distilling code models into speech models is better than distilling speech models. In the experiments of using one teacher, distilling HuBERT performs better than HuBERT itself and distilling any of the code teachers achieves a significant improvement compared to it. This indicates code teachers may contain extra language information compared to the conventional self-supervised speech models.

Secondly, we observe that distilling code teachers complements self-distillation and can achieve further improvement. When self-distillation is applied, distilling code teacher 1 outperforms distilling HuBERT by 0.2/0.8 WER reduction on the ASR task and 0.98 BLEU improvement on the ST task. This proves that improvements achieved by our method can not be achieved by simply distilling speech models.

2.1.5 Summary

We present CoBERT, a self-supervised speech representation learning approach which enables a speech student encoder to learn contextualized latent representations from both speech and code modalities. First of all, we pre-train code models with HuBERT codes to benefit from training in discrete space. Then, we distil the code model into a speech model, which aims at performing a better learning across modalities. The significant improvement on the ST task indicates the representation of CoBERT may

Table 2.3: Comparison of using different teacher models in terms of BLEU and WER scores. For ASR, the models are fine-tuned with 100 hours subset of LibriSpeech.

Teacher model	**BLEU** ↑	**WER** ↓	
	SUPERB ST	test-clean	test-other
W/o self-distillation			
HuBERT	17.19	5.0	10.6
code teacher 1	17.81	4.3	9.8
code teacher 2	18.73	4.2	9.3
With self-distillation			
HuBERT	17.14	4.5	10.4
code teacher 1	18.12	4.3	9.6
code teacher 2	19.10	4.0	8.9

carry more language information compared to prior work.

In this work, only codes generated from the unpaired speech are used to pre-train the code teacher and the amount of them is far less than the data scale to train a state-of-the-art text encoder. Future work may investigate the way to exploit text data to compensate for the paired code scarcity. We would also like to evaluate CoBERT on more spoken language understanding tasks and pre-train a multilingual CoBERT for multilingual speech translation task.

2.2 Encoder-Decoder Pre-training via Pseudo Codes Without Text

2.2.1 Introduction

Self-supervised learning has emerged as an effective paradigm for representation learning, first demonstrating strong impact in natural language processing through models such as BERT [Devlin et al., 2019] and BART [Lewis et al., 2020]. Its central

advantage lies in exploiting large-scale unlabeled data during pre-training so that downstream tasks can be improved with reduced reliance on annotated supervision, an idea that has also been shown beneficial for automatic speech recognition (ASR) [Baevski et al., 2019; Fan et al., 2019; Wang et al., 2020c; Chung and Glass, 2020; Wang et al., 2021a; Jiang et al., 2021].

Within speech self-supervision, existing approaches can be broadly understood through their training objectives as either contrastive learning [Oord et al., 2018; Schneider et al., 2019; Baevski et al., 2020b] or reconstructive learning [Chung and Glass, 2020; Hsu et al., 2021; Chen et al., 2022a]. In the contrastive line, CPC [Oord et al., 2018] adopts a probabilistic contrastive loss that encourages the latent space to retain informative structure, while an autoregressive model is trained to distinguish future frames from negative examples. Wav2vec [Schneider et al., 2019], in turn, pre-trains a simple multi-layer convolutional neural network with a noise-contrastive binary classification objective. In the reconstructive line, APC [Chung and Glass, 2020] uses a unidirectional encoder to reconstruct future frames, thereby learning meaningful, non-specific, and transferable speech representations. HuBERT [Hsu et al., 2021] follows a different strategy: pseudo labels are first produced through offline clustering, and these labels then serve as targets for a BERT-like training loss that can be further improved through iteration.

Despite this progress, prior work has largely concentrated on pre-training the speech encoder for spoken downstream tasks, whereas the decoder in end-to-end encoder-decoder ASR systems typically remains un-pre-trained and therefore continues to depend heavily on large quantities of transcribed audio. Although some studies have explored pre-training a Transformer decoder for end-to-end ASR, these approaches require additional unpaired text data [Fan et al., 2019; Gao et al., 2021; Ao et al., 2021]. This raises a fundamental question: *Can we pre-train the ASR decoder with speech-only data?* Two obstacles make this problem nontrivial. First, speech

is continuous whereas textual representations are discrete. Second, the decoder is designed to generate text, which is substantially different from conventional speech representations such as waveform, log-mel fbank feature, or hidden states.

To address this issue, we propose **Speech2C**, a speech-to-code pre-trained model designed to pre-train an encoder-decoder architecture using speech-only data. The framework adopts multi-task learning and introduces two pre-training objectives built on acoustic units obtained from an offline clustering model, namely pseudo codes. The first objective predicts pseudo codes from the encoder outputs through masked language modeling (MLM), in a manner similar to HuBERT. The second objective trains the decoder to autoregressively reconstruct reduced pseudo codes rather than real text transcriptions. Because both codes and text are discrete representations and both encode semantic information from speech, this formulation provides a bridge between speech-only supervision and text generation. Our observation that a certain portion of the codes is highly correlated with the transcriptions further suggests that, within this pre-training framework, the decoder can learn text prediction.

We conduct massive experiments on the Librispeech dataset to validate Speech2C. To the best of our knowledge, this is the first work to pre-train an encoder-decoder model for ASR with only speech-only data, and the proposed Speech2C achieves a new state-of-the-art performance on the test set of Librispeech.

2.2.2 Related Work

We consider our work most related to HuBERT [Hsu et al., 2021], which benefits from an offline clustering step to provide pseudo labels for a BERT-like pre-training. The backbone of HuBERT includes a convolutional feature encoder and a Transformer context encoder. During pre-training, HuBERT first uses k-means to learn the initial quantizer that maps speech signals to discrete labels, and performs BERT-style pre-training where the inputs are masked speech signals and prediction targets are

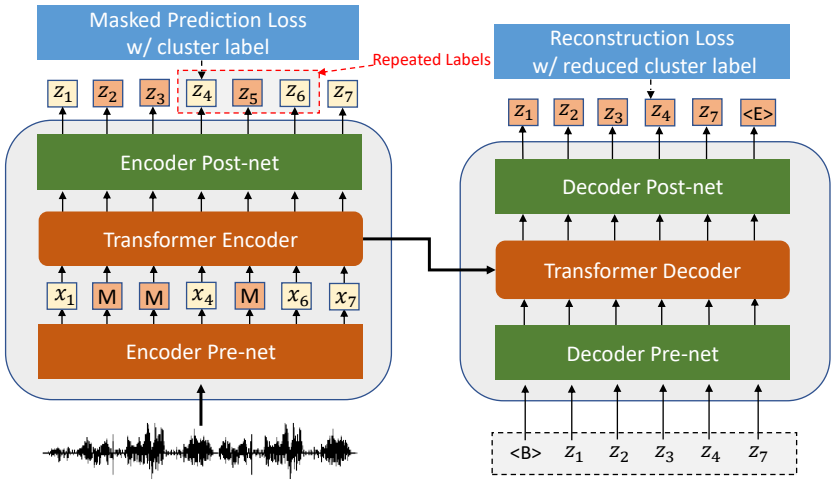


Figure 2.2: The framework of our proposed Speech2C model, which is pre-trained with masked prediction loss and reconstruction loss on unpaired speech data for end-to-end ASR. In this example, z_4, z_5 and z_6 are from the same class and can be reduced to a single label for the decoder. In the fine-tuning stage, we initialize the ASR model with pre-trained Speech2C by removing the encoder post-net and the decoder pre-net/post-net, since they are trained to process pseudo codes.

discrete labels. Moreover, HuBERT allows refinement on the pseudo label by further using the pre-trained model as the new quantizer to train a new iteration of the model. However, HuBERT model only pre-trains a speech encoder, leaving the decoder not pre-trained for the encoder-decoder based tasks, such as end-to-end ASR [Li, 2021]. Based on HuBERT encoder, our proposed Speech2C model can also pre-train a Transformer decoder with pseudo label from the clustering model. Besides, some research works [Lakhotia et al., 2021; Lee et al., 2021] attempt to leverage pseudo codes to reconstruct audios, but they are mainly designed for speech resynthesis and speech to speech translation tasks.

In addition, an idea was explored to pre-train a decoder for end-to-end ASR [Fan et al., 2019; Gao et al., 2021; Ao et al., 2021]. The authors in [Fan et al., 2019] employ a single speaker text to speech (TTS) system to generate synthesized speech from a large number of transcripts, and use the generated speech-text pairs to pre-train the decoder. In [Gao et al., 2021], unpaired text data are used to pre-train the transformer

decoder, which is pre-trained as a conditional language model by constructing empty or artificial states to replace the real encoder hidden states. Leveraging large-scale unpaired speech and text data, SpeechT5 [Ao et al., 2021] pre-trains a shared encoder-decoder model for various spoken language tasks. However, all previous work still utilize text data to pre-train a decoder for end-to-end ASR. In contrast, Speech2C is the first work to pre-train a Transformer decoder for ASR without text data.

2.2.3 Methods

In this section, we first illustrate our model architecture, based on which, we present two pre-training tasks in our proposed Speech2C, the masked prediction loss for the encoder and the pseudo-code reconstruction loss for the decoder.

2.2.3.1 Model Architecture

The model architecture of the proposed Speech2C for pre-training is composed of an encoder network that extracts latent speech representations from raw acoustic inputs and learns contextualized speech representations, and a decoder network for autoregressively reconstructing the pseudo codes of corresponding source speech. Both encoder network and decoder network are boosted with relative positional encoding [Shaw et al., 2018]. Figure 2.2 illustrates the framework of the proposed Speech2C.

The encoder network follows the HuBERT Base architecture, with an encoder pre-net, the Transformer encoder, an encoder post-net, as shown on the left panel of Figure 2.2. More specifically, the encoder pre-net is a convolutional network for pre-processing waveform, which is of seven 512-channel layers with strides [5,2,2,2,2,2,2] and kernel widths [10,3,3,3,3,2,2]. The Transformer encoder contains 12 layers with model dimension 768, inner dimension 3072 and 12 attention heads. Moreover, the encoder-post network contains a projection layer and a code embedding layer, which are utilized to convert hidden states into pseudo codes.

The decoder network also contains a decoder pre-net, the Transformer decoder, and a decoder post-net. The pre-net transforms a code index into an embedding vector. The Transformer decoder has a similar architecture to the Transformer encoder except for the cross-attention and the masked self attention. Finally, the post-net transforms the hidden state into the probability distribution of codes, normalized by the softmax function.

2.2.3.2 Self-Supervised Speech Pre-Training

The proposed pre-training learning method for Speech2C has access to speech-only data. We introduce two pre-training tasks to pre-train the encoder-decoder model, including masked prediction loss for the encoder and reconstruction loss for the decoder.

Masked prediction loss. During pre-training, the encoder pre-net first generates a feature sequence $\mathbf{x}$ from waveform by down-sampling, then the Transformer encoder consumes masked acoustic features $\mathbf{\tilde{x}}$ and output hidden states $\mathbf{h}^L$. Furthermore, the network including encoder post-net is optimized to predict the discrete target sequence $\mathbf{z}$, where each $z_t \in [C]$ is a C-class categorical variable. The distribution over codewords is parameterized with:

$$p_f(c|\mathbf{x}, t) = \frac{\exp(\text{sim}(\mathbf{h}_t^L \mathbf{W}, \mathbf{e}_c)/\tau)}{\sum_{c'=1}^{C} \exp(\text{sim}(\mathbf{h}_t^L \mathbf{W}, \mathbf{e}_{c'})/\tau)} \tag{2.7}$$

where $\mathbf{W}$ is a projection matrix, $\mathbf{h}_t^L$ is the output hidden state for step t and layer L, $\mathbf{e}_c$ is the embedding for codeword c, sim(a, b) computes the cosine similarity between two vectors and $\tau = 0.1$ is used to scale the logit.

Specifically, $\mathbf{\tilde{x}}$ comes from $\mathbf{x}$ by span mask strategies, where 8% of timesteps are randomly selected as start indices, and spans of 10 steps are masked. Based on the above distribution, we denote the cross-entropy loss computed over masked timesteps

as

$$\mathcal{L}_{mlm} = \sum_{t \in \mathcal{M}} \log p_f(\mathbf{z}_t | \tilde{\mathbf{x}}, t), \tag{2.8}$$

where, $\mathcal{M}$ denotes the set of masked timesteps, and $\mathbf{z}_t$ denotes the frame-level target at timestep t from $\mathbf{Z}$.

Pseudo-code reconstruction loss. In addition to masked prediction loss in the encoder, we also design a reconstruction loss to pre-train the Transformer decoder. Following the denoising autoencoder in BART [Lewis et al., 2020], the decoder network is optimized to generate the reduced pseudo codes with the maximum likelihood estimation as

$$\mathcal{L}_{mle} = \sum_{n=1}^{N} \log p(\mathbf{z}_n | \mathbf{z}_{<n}, \tilde{\mathbf{x}}), \tag{2.9}$$

where N is the length of pseudo codes.

Pseudo codes are similar to real texts because (1) they are discrete representations and have fixed vocabulary; (2) they all have rich semantic information that can be aligned to speech fragments. Hence, we believe this pre-training method on pseudo codes can help the decoder learn how to generate text sequences. The pseudo codes of adjacent speech frames have some repeated codes, which may represent similar semantic information, but repeated words are rarely used consecutively in textual languages. To reduce the gap between pseudo code and real text, we remove the repeating code of adjacent speech frame, which will be studied in the ablation study.

2.2.4 Experiments

2.2.4.1 Training Details

All models are implemented in Fairseq[5] [Ott et al., 2019]. For speech pre-training, we use the full 960 hours of LibriSpeech [Panayotov et al., 2015a] audio without tran-

[5] https://github.com/pytorch/fairseq

scription. We optimize the model with Adam [Kingma and Ba, 2014] by warming up the learning rate for the first 8% of updates to a peak of 2×10^{-4}, which is linearly decayed for the following updates. We pre-train the proposed Speech2C model on 32 V100 GPUs with a batch size of around 87.5s samples per GPU for 400k steps.

Table 2.4: WER on the LibriSpeech test sets when training on the 10 hours and 100 hours subset. † indicates that the results are not reported in [Hsu et al., 2021] and obtained by fine-tuning the public released model.

Model	LM	test-clean	test-other
10 hours subset			
wav2vec2.0 BASE [Baevski et al., 2020b]	None	11.1	17.6
HuBERT BASE † [Hsu et al., 2021]	None	10.1	16.8
Our Speech2C	None	**7.8**	**13.1**
wav2vec2.0 BASE [Baevski et al., 2020b]	4-gram	4.3	9.5
wav2vec2.0 BASE [Baevski et al., 2020b]	Transf	3.2	7.8
HuBERT BASE [Hsu et al., 2021]	4-gram	4.3	9.4
Our Speech2C	Transf	**3.1**	**7.0**
100 hours subset			
wav2vec2.0 BASE [Baevski et al., 2020b]	None	6.1	13.3
wav2vec2.0 LARGE [Baevski et al., 2020b]	None	4.7	9.0
HuBERT BASE † [Hsu et al., 2021]	None	6.3	13.2
SpeechT5 [Ao et al., 2021]	None	4.4	10.4
Baseline	None	5.0	11.9
Our Speech2C	None	**4.3**	**9.0**
wav2vec2.0 BASE [Baevski et al., 2020b]	4-gram	3.4	8.0
wav2vec2.0 BASE [Baevski et al., 2020b]	Transf	2.6	6.3
HuBERT BASE [Hsu et al., 2021]	4-gram	3.4	8.1
SpeechT5 [Ao et al., 2021]	Transf	2.4	5.8
Baseline	Transf	2.5	6.3
Our Speech2C	Transf	**2.4**	**5.2**

For the fine-tuning, we employ 10 hours and 100 hours as the supervised paired corpus, and use the character set as the model units for the text. Because pseudo codes and textual characters have different vocabulary, we initialize the ASR model

with pre-trained Speech2C without the encoder post-net and decoder pre-net/post-net, which are trained from scratch in our fine-tuning model. We utilize the CTC and cross-entropy loss to fine-tune the model [Watanabe et al., 2017], where the loss weights are 0.5 for both of them. The models are trained on 16 v100 GPUs with a batch size of 100s samples per GPU. The learning rate is warmed up for the first 10% steps, held as a constant for the following 40% steps, and is decayed linearly for the rest steps. For the 10/100 hours subset, we train the model with a learning rate of 2e-5/4e-5 for 25k/80k, and fix the encoder part for the first 10k/25k steps.

For ASR inference, we apply the joint CTC/attention decoding [Hori et al., 2017] and train a transformer language model (LM) by LibriSpeech-LM Corpus with the same architecture as in [Ao et al., 2021] for the shallow fusion [Gulcehre et al., 2015] to boost the performance of our baseline and Speech2C. We sweep over the weights of the language model and CTC from 0 to 1 on the dev-other subset and choose the best weights according to the WER.

2.2.4.2 Main Results

The results of ASR on the 10 hours and 100 hours set of LibriSpeech are reported in Table 2.4. The WER is evaluated on the standard Librispech test-clean/other sets. We compare with several state-of-the-art self-supervised approaches, including encoder-based wav2vec 2.0 [Baevski et al., 2020b] and HuBERT [Hsu et al., 2021], and encoder-decoder based SpeechT5 [Ao et al., 2021], which utilizes the unpaired speech and text corpus to pre-train. We build a strong encoder-decoder based ASR baseline system, which has the same model structure as our Speech2C, while the weights of the encoder are initialized by the HuBERT Base model.

Without LM fusion, the baseline outperforms wav2vec 2.0 Base and HuBERT Base with the help of the joint CTC/attention decoding, which shows the importance of the decoder. Our proposed Speech2C model achieves significant improvements on

all settings compared to wav2vec 2.0 BASE, HuBERT BASE, SpeechT5 [Ao et al., 2021] and our strong baselines, demonstrating the superiority of the proposed pre-training method. More specifically, our Speech2C without LM gets a relative 19.2% WER reduction on the average of all sets compared to the baseline system for 100h subset, which achieves state-of-the-art performance. Furthermore, when decoding with LM shallow fusion, our Speech2C still obtains the lower WERs than the strong baseline on all sets.

2.2.4.3 Ablation Study

We present a series of ablation studies in the following sections to learn how code reduction, continuing pre-training, and model layer numbers affect performance. The models for ablation studies are pre-trained on 960 hours and fine-tuned on the 100-hour subset using fixed hyperparameters.

Effect of code reduction. Here, we start with probing the effectiveness of the k-means clustering algorithm concerning code reduction. In Table 2.5, we summarize the results of Speech2C with repeated pseudo codes when calculating reconstruction loss, and the Speech2C with repeated codes performs slightly worse than Speech2C with reduced codes. Moreover, reducing repeated codes has the following advantages: (1) the average length of reduced codes is significantly shorter than that of repeated codes, which will accelerate the training progress of Speech2C; (2) removing repeated codes does not loss semantic information, and brings pseudo codes closer to the text.

Continued pre-training from released model. In addition to pre-train Speech2C from scratch, our method can also support continually pre-train from a pre-trained speech encoder model, such as pre-trained HuBERT. Table 2.6 shows the experimental results of two pre-trained Speech2C. Although two pre-training methods achieve

Table 2.5: Comparison of Speech2C with repeated codes or reduced codes in terms of average length and WER.

Model	Length	test-clean	test-other
Speech2C (repeated)	358	4.4	9.4
Speech2C (reduced)	216	4.3	9.0

comparable performance, initializing the encoder of Speech2C with HuBERT can speed up the convergence and reduce the training time.

Table 2.6: WER scores of Speech2C trained from scratch or initialized by HuBERT encoder.

Model	test-clean	test-other
Speech2C (from scratch)	4.3	9.0
Speech2C (from HuBERT)	4.1	9.1

Effect of layer numbers. Compared to wav2vec and HuBERT which adopt CTC inference based on speech encoder, the encoder-decoder based ASR (eg., SpeechT5 and Speech2C) adds an external decoder to generate text. To reduce the influence of model parameters, we reduce the model layers of encoder-decoder model to the similar parameters of encoder model. As shown in Table 2.7, we list the experimental results of Speech2C with different encoder and decoder layers. We change the default setting of 12 encoder layers and 6 decoder layers to the total same layer number of wav2vec2.0 BASE and HuBERT BASE, such as 10 encoder layers and 2 decoder layers. Results show that although reducing the model layers degrade the performance of Speech2C, it still behaves better than encoder based ASR model.

Table 2.7: WER scores of Speech2C with different layer numbers.

Model	Size	Layer (enc-dec)	test-clean	test-other
wav2vec2.0	95M	12-0	6.1	13.3
HuBERT	95M	12-0	6.3	13.2
SpeechT5	154M	12-6	4.4	10.4
Speech2C	104M	8-4	4.9	10.9
Speech2C	100M	10-2	4.6	9.8
Speech2C	152M	12-6	4.3	9.0

2.2.4.4 Analysis

In this section, we want to answer a question: why pre-train the decoder with pseudo code can help text generation in ASR? Frames of the same phonemes are more likely labeled as similar sequences of pseudo codes. We give an example chosen from the LibriSpeech dataset to show the high relevance of pseudo codes to text data and the patterns inherent in them. As shown in Figure 2.3, the codes corresponding to "wonder" in different samples are also similar and have an obvious pattern, which demonstrates that pseudo codes can be regarded as an intermediary language between waveform and transcription.

2.2.5 Summary

This work proposes a novel model, Speech2C, to pre-train an encoder-decoder model with speech-only data. We present two pre-training tasks including masked prediction task and reconstruction task by taking advantage of acoustic units, i.e. pseudo codes derived from an offline clustering model. Massive experiments and analyses on the LibriSpeech dataset show the effectiveness and superiority of our proposed Speech2C. To the best of our knowledge, Speech2C is the first work to pre-train an encoder-decoder based ASR model with speech-only data. For future work, we will

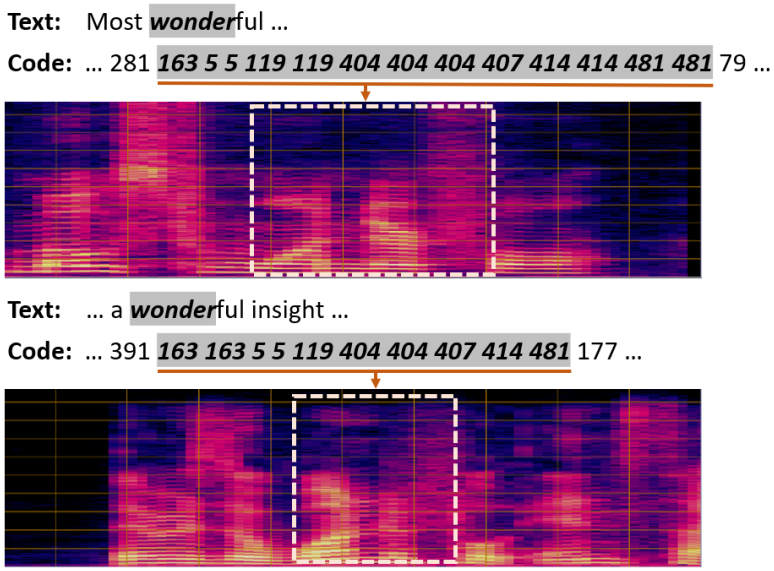


Figure 2.3: The transcripts, pseudo codes and Mel-Spectrum from two sentences, where the white boxes and the bold pseudo codes corresponding to the subword "wonder".

pre-train a multilingual Speech2C to address cross-lingual tasks, such as speech translation.

□ End of chapter.

Chapter 3

Robust Speech Modeling in Multi-Speaker Real-World Scenarios

Summary

This chapter addresses multi-speaker mixture speech from two complementary perspectives. Section 3.1 presents SA-WavLM, a speaker-aware self-supervised pre-training model that leverages speaker information to handle mixture speech through an "extract-merge-predict" pipeline. Section 3.2 proposes USED, a unified model for universal speaker extraction and diarization that jointly addresses both tasks for speech mixtures with varying overlap ratios and variable numbers of speakers.

Building on the speech-only representation learning framework established in Chapter 2, this chapter moves from generic pre-training to robust speech modeling in realistic acoustic conditions. While the previous chapter focused on learning effective representations from unlabeled speech, practical speech systems must operate in en-

vironments that are far more challenging than clean single-speaker settings. In real-world scenarios, speech is often affected by overlap, interference, and speaker ambiguity, which place additional demands on both representation learning and downstream modeling.

To address this problem, this chapter studies robust speech modeling in multi-speaker real-world scenarios. It first examines speaker-aware self-supervised learning for mixture speech, which extends the idea of representation learning to acoustically entangled multi-speaker inputs. It then presents a unified framework for speaker extraction and diarization, moving from representation-level robustness to task-level modeling under realistic overlap conditions. In this sense, this chapter forms a bridge between foundational speech representation learning and more practical speech systems designed for complex real-world environments.

3.1 SA-WavLM: Speaker-Aware Self-Supervised Pre-training for Mixture Speech

3.1.1 Introduction

In recent years, self-supervised learning (SSL)-based pre-training has advanced substantially [Mohamed et al., 2022; Ericsson et al., 2022; Liu et al., 2023b]. By exploiting large-scale unlabeled speech data, such models learn general speech representations that are useful across a broad set of downstream tasks, including speech recognition, speaker verification, and acoustic word embeddings [Schneider et al., 2019; Chen et al., 2022d; Lin et al., 2023]. Despite this progress, many influential pre-trained models, such as wav2vec 2.0 [Baevski et al., 2020b] and HuBERT [Hsu et al., 2021], are developed primarily for clean speech, which limits their effectiveness when the input consists of mixture speech.

Mixture speech, in which multiple speakers may talk simultaneously, constitutes a particularly challenging condition. An early effort to address this issue is WavLM [Chen et al., 2022a], which is trained on simulated mixture speech with the objective of denoising and performing masked prediction for the corresponding clean speech. In this formulation, the simulated mixtures are constrained so that the mixing portion remains below 50%, and a primary speaker is identified according to longer speech duration. Pre-training then focuses only on masked prediction for that primary speaker, while other interfering speakers are ignored. Related approaches, including [Zhu et al., 2023b; Wang et al., 2022], similarly define the primary speaker as the solo speaker in the presence of non-speech background noise. As a result, the learned representations retain information for only one speaker. Although such designs improve the robustness of pre-trained models to mixture speech, later studies have indicated that they remain fundamentally oriented toward single-speaker speech and may therefore be suboptimal for mixture-speech tasks [Fazel-Zarandi and Hsu, 2023], where all speakers are equally important.

From the perspective of the cocktail party problem [Qian et al., 2018; Cherry, 1953], the ability to process mixture speech is essential, which motivates the development of pre-trained models specifically suited to this setting. Recent studies have accordingly extended SSL pre-training toward mixture speech. In [Wang et al., 2023d], both single-speaker speech and two-speaker mixtures are simulated during pre-training, and the model is trained to predict masked timestamps for all clean speeches from the mixture input. This idea is further generalized in [Fazel-Zarandi and Hsu, 2023] to mixtures containing more speakers through the use of a permutation invariant training (PIT) loss [Yu et al., 2017]. These approaches explicitly construct pre-training schemes for learning mixture representations from mixture inputs, and because they treat all speakers in the mixture as relevant, they substantially strengthen the resulting models' ability to handle mixture speech.

Alongside these developments, another promising direction is to incorporate additional speaker information. Speaker information has been shown to be effective in a range of related problems, including speaker-attributed automatic speech recognition (ASR) [Kanda et al., 2021b, 2022] and speaker diarization (SD) [Medennikov et al., 2020]. For example, [Kanda et al., 2021b] jointly performs speech recognition and speaker identification in multi-speaker scenarios, while [Medennikov et al., 2020] estimates the speech activities of all speakers in an input mixture given each speaker's profile. Taken together, these studies demonstrate that integrating speaker information into the model is effective for addressing the challenges posed by mixture speech.

Motivated by these observations, this dissertation introduces SA-WavLM, a speaker-aware self-supervised pre-trained model designed to better handle mixture speech. Unlike WavLM [Chen et al., 2022a], which considers only a primary speaker and may therefore neglect others, SA-WavLM is designed to address all speakers present in the input mixture. It adopts an "extract-merge-predict" pre-training pipeline. In the "extract" stage, SA-WavLM derives an individual representation for each speaker in the mixture, conditioned on the corresponding speaker information. This speaker information is provided through speaker embeddings and incorporated into the model via conditional layer normalization. In the "merge" stage, the individual representations are combined and interactions across speakers are modeled through a Speaker Merge Block. In the final "predict" stage, SA-WavLM predicts pseudo labels for each speaker. In addition, a speaker shuffling strategy is introduced to ensure invariance to speaker order and speaker absence. This design removes the need for the mixture-simulation constraints used in WavLM and improves interference removal, including cases in which the target speaker is not the primary speaker in the mixture.

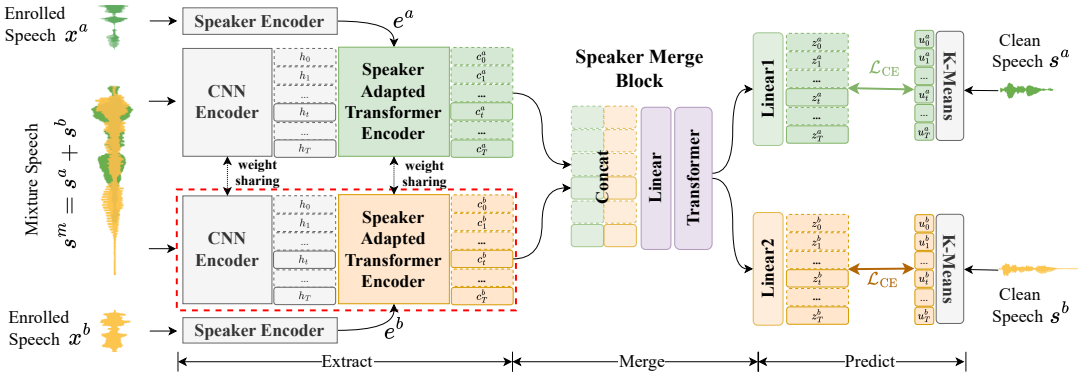


Figure 3.1: The overview of SA-WavLM architecture and the "extract-merge-predict" pipeline. Given the input mixture speech, the proposed model first extracts the individual representations for each speaker in the input. Subsequently, the Speaker Merge Block merges the individual representations and models their interactions before making the final prediction. In downstream tasks, only the "extract" stage is used, which is the part within the red dotted box.

3.1.2 WavLM

Our work is most related to WavLM [Chen et al., 2022a], a self-supervised speech pre-trained model that combines denoising and masked speech prediction during pre-training. WavLM consists of a convolutional neural network (CNN) $\mathcal{F}(\cdot)$ and a Transformer encoder $\mathcal{G}(\cdot)$. Given a mixture speech s^m simulated from the clean speeches s^a of speaker a and s^b of speaker b, WavLM first extracts the frame-level speech representations $H^m = [h_1^m, h_2^m, ..., h_T^m]$ through $\mathcal{F}(\cdot)$. These representations are the passed through a partial masking process $\mathcal{M}(\cdot)$ and fed into $\mathcal{G}(\cdot)$. Let us assume speaker a is the primary speaker whose speech s^a is longer than s^b. WavLM performs denoising for speaker a to get the contextual representations $C^a = [c_1^a, c_2^a, ..., c_T^a]$ that corresponds to speech s^a. The whole pipeline can be formalized as

$$C^a = [c_1^a, c_2^a, ..., c_T^a] = \mathcal{G}\left[\mathcal{M}(\mathcal{F}(s^m))\right] \tag{3.1}$$

In the masked prediction process, the generated C^a should be able to predict the

pseudo labels $U^a = [u_1^a, u_2^a, ..., u_T^a]$ for the masked frames in s^a. The masked speech prediction loss during the pre-training is the cross-entropy (CE)

$$\mathcal{L}_m = \mathcal{L}_{\text{CE}}(C^a, U^a) = \sum\nolimits_{t \in O} \log\, p_t(\, u_t^a \,|\, c_t^a) \tag{3.2}$$

where O denotes the set of masked indices in H^m. An offline clustering model (i.e. k-means) is used to get the pseudo labels.

In Eq. (3.1), s^a is assumed to have a longer speech duration in s^m. In fact, during the mixture simulation for pre-training WavLM, a constraint is applied such that the mixing portion must be less than 50%. This constraint ensures that the speech from the primary speaker is always longer than the other speech, thereby informing the model about the primary speaker (to retain) and the interfering speaker (to remove). In practice, alternative primary speaker definitions, such as the speaker with higher energy or the solo speaker in noisy speech (clean speech with background non-speech noises), are viable too.

3.1.3 Proposed SA-WavLM

In WavLM, only the primary speaker is concerned. However, the definition of a primary speaker may not align with multi-speaker speech tasks, including speech diarization, speech separation, and multi-speaker speech recognition, where all speakers in the mixture speech are equally important.

Building upon WavLM, we propose SA-WavLM to address this limitation. Figure 3.1 illustrates the overview of SA-WavLM architecture and the "extract-merge-predict" pipeline. Given the mixture speech $s^m = s^a + s^b$, where s^a and s^b are two clean speeches from speakers a and b, we first "extract" representations of speakers a and b individually by the Speaker Adapated Transformer Encoder (SATE) denoted as $\mathcal{G}_{\text{SATE}}(\cdot)$, replacing the $\mathcal{G}(\cdot)$ in WavLM. The individual representations are then

"merged" using a Speaker Merge Block (SMB), which facilitates their interactions. Finally, the training objective of SA-WavLM is to "predict" the pseudo labels of speeches s^a and s^b. In addition, to ensure the model's invariance towards speaker order and absence, we introduce a speaker shuffling strategy.

3.1.3.1 Speaker Adapted Transformer Encoder (extract)

Speaker Adapted Transformer Encoder (SATE) is designed to extract the contextual representation for the target speaker. Given speaker embedding e^k of the target speaker $k \in \{a, b\}$ and masked frame-level representation $H^m = [h_1^m, h_2^m, ..., h_T^m]$ of s^m extracted by $\mathcal{F}(\cdot)$, SATE, denoted by $\mathcal{G}_{\text{SATE}}$, is trained to extract the contextual representation $C^k = \left[c_1^k, c_2^k, ..., c_T^k\right]$ that corresponds to s^k. This is formulated as

$$C^k = \left[c_1^k, c_2^k, ..., c_T^k\right] = \mathcal{G}_{\text{SATE}}\left[\mathcal{M}(\mathcal{F}(s^m)), e^k\right], k \in \{a, b\} \tag{3.3}$$

SATE consists of a Speaker Adapated Transformer Layer (SATL) and multiple Vanilla Transformer Layers (VTLs). Here the VTL in SATE has the same structure as the Transformer layer in the Transformer encoder $\mathcal{G}(\cdot)$ of WavLM. The main extraction process of SATE is performed through SATL, which replaces the traditional layer normalization in VTL with conditional layer normalization (CLN). In VTL, the layer normalization operation is given by

$$H_{\text{VTL}} = \frac{H^m - E\left[H^m\right]}{\sqrt{Var\left[H^m\right] + \epsilon}} \otimes \gamma + \beta, \tag{3.4}$$

where $E\left[H^m\right]$ and $Var\left[H^m\right]$ are the mean and variance of the input representation H^m, and γ and β are learnable weight and bias for applying the element-wise affine transformation.

To extract the representations of speech s^k given e^k, the CLN in SATL replaces γ

with speaker-specific scaling, i.e.

$$H_{\text{SATL}} = \frac{H^m - E\left[H^m\right]]}{\sqrt{Var[H^m] + \epsilon}} \otimes \left[w(e^k) \cdot \gamma + \theta(e^k)\right] + \beta, \tag{3.5}$$

where $w(\cdot)$ and $\theta(\cdot)$ are linear projections to project the speaker embedding e^k to the feature dimension of H^m to perform the element-wise multiplication. In fact, CLN modulates the normalization output via a specific speaker's latent representation.

3.1.3.2 Speaker Merge Block (merge)

Motivated by TS-VAD [Medennikov et al., 2020], it is believed that interaction between different speakers is important for better extraction and separation abilities. Therefore, in SA-WavLM, we design a Speaker Merge Block (SMB) to allow interaction between different speakers. The structure of SMB is also shown in Figure 3.1.

SMB first concatenates C^a and C^b of speaker a and b, extracted respectively by SATE as shown in Eq. (3.3), along the feature dimension. Later, the concatenated representations are projected down by a linear layer. A single VTL, which has the same structure as the VTL in SATE, is deployed to model the down-projected representations and their interactions, generating the mixture contextual representations C^m

$$C^m = \text{VTL}(\text{Linear}(\text{Concat}(C^a, C^b))) \tag{3.6}$$

Note that the SMB is used during pre-training only to facilitate the interactions. In downstream applications, SMB is removed and only CNN and SATE are used.

3.1.3.3 Prediction and training objective (predict)

Unlike Cocktail HuBERT [Fazel-Zarandi and Hsu, 2023] where PIT [Yu et al., 2017] is needed to find the optimal alignment between the predictions and pseudo labels, SA-

WavLM determines the order of predictions based on the order of speaker embeddings injected. That is, if the concatenation order in Eq. (3.6) is a followed by b, Linear1 predicts the pseudo labels of clean speech s^a and Linear2 predicts that of clean speech s^b and vice versa. Here, the former order, i.e. a followed by b, is used for illustration

$$Z^a = \text{Linear1}(C^m), \quad Z^b = \text{Linear2}(C^m) \tag{3.7}$$

The final masked speech prediction loss is the summation of Cross Entropy (CE) losses for the two speakers

$$\mathcal{L}_m = \sum_{k\in\{a,b\}} \mathcal{L}_{\text{CE}}(Z^k, U^k) = \sum_{k\in\{a,b\}} \sum_{t\in O^k} \log\, p_t(\, u_t^k \mid z_t^k) \tag{3.8}$$

3.1.3.4 Speaker shuffling strategy

To keep the model invariant to speaker order, we deploy a speaker shuffling strategy to shuffle the order of speaker embeddings and the corresponding pseudo labels. In our mixture simulation, we have two-speaker scenarios and one-speaker scenarios. Two-speaker scenarios include two-speaker overlapped speech $s^m = s^a + s^b$ and noisy two-speaker overlapped speech $s^m = s^a + s^b + n$, where n is the background noise. One-speaker scenarios include clean speech s^a and noisy single-speaker speech $s^n = s^a + n$. For two-speaker scenarios, the speaker embeddings e^a and e^b are used directly. While for one-speaker scenario, we will use speaker embeddings e^a and, either 1) e^c from a random speaker that is different from speaker a, with a probability of α, or 2) e^s, a learnable vector indicating non-speaker existence, with a probability of $(1-\alpha)$. Subsequently, we shuffle the order of all the speaker embeddings injected into SATE, which determines the order of the Concat operation in Equation 3.6. This strategy aims to enhance the model's insensitivity towards the speaker order and robustness towards the speaker absence, thereby improving the accuracy of extraction.

The pseudo labels corresponding to e^c or e^s will be S, which is a vector of silence tokens.

3.1.4 Experimental Setups

3.1.4.1 Training datasets

Following [Wang et al., 2023d], we pre-train our model on the data simulated from the 960-hour LibrSpeech [Panayotov et al., 2015a] and the DNS Challenge's noise datasets [Reddy et al., 2020]. Due to limited computational resources, only one- and two-speaker scenarios are simulated. This includes clean speech, noisy single-speaker speech, two-speaker overlapped speech, and noisy two-speaker overlapped speech, which are generated through a dynamic mixing strategy [Zhang and Qian, 2023]. The enrolled speech of the target speaker is randomly selected from the LibriSpeech corpus, and it should be different from the one used to simulate the mixture speech. The speaker embedding is generated from the selected enrolled speech using the pre-trained CAM++ [Wang et al., 2023c]. All speeches are sampled at 16kHz.

3.1.4.2 Models and training details

All pre-training experiments are conducted using Fairseq toolkit [Ott et al., 2019]. Our model is based on WavLM Base [Chen et al., 2022a], while replacing a Vanilla Transformer Layer (VTL) with a Speaker Adapted Transformer Layer (SATL) and adding a Speaker Merge Block (SMB). SATL can replace any VTL in the Transformer encoder, but here we only replace the first layer. The parameters of CNN encoder and SATE are initialized from the pre-trained WavLM Base, except for w and θ of SATL. For the SMB, the weights are randomly initialized. Training takes 400k steps with a learning rate of 7e-5, using the pseudo labels generated from the 9th transformer layer of the HuBERT Base model [Fazel-Zarandi and Hsu, 2023]. The probability α used in

the speaker shuffling strategy is set to 0.5. Other hyperparameters are consistent with those of the WavLM Base. Note that although SA-WavLM has the largest model size as shown in Table 3.1, the additional SMB is removed for downstream tasks. This leaves our model size with 94.97M parameters, which is comparable to other pre-trained models (e.g. WavLM Base, HuBERT Base etc).

3.1.4.3 Downstream evaluations

To evaluate the effectiveness of our models on the cocktail party problem, we evaluated our models on different mixture speech tasks including speech enhancement (SE), separation (SS), diarization (SD), extraction (Ext) and multi-speaker speech recognition (ASR). For SE, the dataset used is Voicebank-DEMAND [Veaux et al., 2013]. For the remaining, the different subsets of Libri2Mix are used. In particular, SD uses the "mix-both max mode", ASR uses the "mix-clean max mode" and SS and Ext use the "mix-clean min mode". All the datasets in downstream evaluations have 16kHz sample rates.

As baselines, we select state-of-the-art pre-trained models, including wav2vec 2.0 [Baevski et al., 2020b], HuBERT [Hsu et al., 2021], WavLM [Chen et al., 2022a], Wang et al. [Wang et al., 2023d], and Cocktail HuBERT [Fazel-Zarandi and Hsu, 2023]. For all baselines, Base sizes comparable to SA-WavLM are used. Among them, wav2vec 2.0 and HuBERT are trained on clean speech only, while the rest are trained on mixture speech.

3.1.5 Experiments

3.1.5.1 Results on SUPERB benchmark

We evaluate on SUPERB [wen Yang et al., 2021], a well-known benchmark that provides unified evaluations across various speech processing tasks. It offers lightweight

downstream networks that make it easy to compare and draw insights across different pre-trained models. Assessments are made for the provided mixture speech tasks: SE, SS and SD. We report Perceptual evaluation of speech quality (PESQ) and short-time objective intelligibility (STOI) for SE, scale-invariant signal-to-distortion ratio improvement (SI-SDRi) for SS, and the diarization error rate (DER) for SD. In all the experiments, the pre-trained models are frozen and only the downstream networks are fine-tuned.

Table 3.1 lists the evaluation results on the SUPERB benchmark. It can be observed that wav2vec 2.0 and HuBERT do not generalize well to the mixture speech tasks. With the denoising strategy, WavLM outperforms wav2vec 2.0 and HuBERT across all three tasks. Both Wang et. al and Cocktail HuBERT (referred to as C-HuBERT in Table 3.1) are designed for mixture speech, hence demonstrating superior performance on all three tasks. When more speakers are considered during pre-training, C-HuBERT shows further improvement on SS. Compared against all baselines, SA-WavLM either matches or outperforms across all the tasks. Notably, both Wang et.al and SA-WavLM are pre-trained with 2-speaker mixture speech only, yet SA-WavLM demonstrates better performance on SS, surpassing Wang et. al by 0.54dB SI-SDRi. This proves the effectiveness of the use of speaker cues and modeling of speaker interactions.

3.1.5.2 Multi-speaker speech recognition

Following the utterance group-based evaluation settings in [Huang et al., 2023], we evaluate SA-WavLM's performance on multi-speaker ASR. The character-level connectionist temporal classification (CTC) loss is used to fine-tune the pre-trained models except for the CNN encoder. We consider two settings: one without and one with speaker embeddings. In the former, permutation invariant training (PIT) [Yu et al., 2017] is used. In the latter, the same CLN operation as in SA-WavLM is applied to

Table 3.1: Universal representation evaluation on SUPERB.

Methods	#Param	SE		SS	SD
		PESQ↑	STOI↑	SI-SDRi↑	DER↓
FBANK	–	2.55	93.60	9.23	10.05
Wav2vec2.0 Base [Hsu et al., 2021]	95.04M	2.55	93.9	9.77	6.08
HuBERT Base [Hsu et al., 2021]	94.68M	2.58	93.90	9.36	5.88
Wang et.al [Wang et al., 2023d]	95.02M	–	–	10.59	–
C-HuBERT Base [Fazel-Zarandi and Hsu, 2023]	96.00M	**2.63**	94.00	11.08	2.77
WavLM Base [Chen et al., 2022a]	94.70M	2.58	94.00	10.37	4.55
SA-WavLM (Ours)	103.7M	2.62	**94.18**	**11.13**	**1.88**

integrate speaker embeddings.

Table 3.2 reports the word error rate (WER) for all the pre-trained models, with and without the 4-gram language model (LM). For models trained using PIT, concatenated minimum-permutation word error rate (cpWER) [Watanabe et al., 2020] is reported. As shown in Table 3.2, WavLM, designed for mixture speech, obtains a much better performance compared to HuBERT which is designed for clean speech. When incorporating the speaker embeddings into the pre-trained models, both HuBERT and WavLM demonstrate reduced WER in settings without LM. Compared to WavLM, SA-WavLM achieves a more significant relative WER reduction of 37.4%. This can be attributed to SA-WavLM's consideration of speaker interactions, which improves its speaker discrimination ability.

3.1.5.3 Speech extraction and separation

While SUPERB offers a standard and comprehensive benchmark for the research community, many works have investigated to evaluate SSL models' capabilities in different ways, striving to push the performance boundaries further [Chen et al., 2022d; Baevski et al., 2020b; Hsu et al., 2021; Fan et al., 2021; Morais et al., 2022]. However,

Table 3.2: Results on multi-speaker speech recognition.

Pre-trained Model	**Spk. Embedding**	**WER (%)**	
		w/o LM	**w/ LM**
HuBERT Base [Hsu et al., 2021]	✗	22.70	15.60
	✓	17.42	14.88
WavLM Base [Chen et al., 2022a]	✗	15.97	10.38
	✓	13.30	11.36
SA-WavLM (Ours)	✓	**8.39**	**6.49**

the ability of the pre-trained model on speech extraction and separation is relatively under-explored. In [Shi et al., 2021], discretization is first performed on the representations obtained from the frozen pre-trained model and the clean waveform is resynthesized from the discrete tokens. In our experiment, we use the representations from all layers of the frozen pre-trained models directly, as discretization could potentially result in the loss of information. The pre-trained representations of different layers are weighted-sum, upsampled and then concatenated with the supervised model's encoded features before feeding into separation/extraction modules in these models.

Table 3.3 shows the results for speech extraction using BSRNN [Yu et al., 2023] that is integrated with different pre-trained models. The results indicate that BSRNN benefits from integrating with the pre-trained speech representations. SA-WavLM stands out among all the pre-trained models, achieving 3.74dB SDRi and 4.29dB SI-SDRi improvements from BSRNN (8ms).

Table 3.4 presents the results for speech separation using ConvTasNet [Luo and Mesgarani, 2019]. In this experiment, we further investigate the effectiveness of the pre-trained models in low-resource scenarios (e.g. 1%). Consistent with the find-

Table 3.3: Results on speech extraction. The default stride for BSRNN is 8ms. We changed it to 5ms so that BSRNN's features can be aligned with the pre-trained features of 20ms stride.

Extraction Model	**Stride**	**Pre-trained Model**	**SDRi↑**	**SI-SDRi↑**
BSRNN [Yu et al., 2023]	8ms	–	14.0	13.01
	5ms	–	12.39	10.06
		HuBERT Base [Hsu et al., 2021]	14.83	14.11
		WavLM Base [Chen et al., 2022a]	14.64	15.47
		SA-WavLM (Ours)	**17.74**	**17.3**

ings in Table 3.3, ConvTasNet benefits from the pre-trained representations across all settings (i.e., 1%, 10% and 100%). The benefits are particularly prominent in low-resource scenarios. This can be explained as pre-trained representations provide more noise-invariant contextual information. Again, among all the pre-trained models, SA-WavLM demonstrates the most promising performances.

3.1.6 Summary

This section proposed SA-WavLM, a novel speaker-aware self-supervised pre-trained model for mixture speech. Pre-trained with the designed "extract-merge-predict" pipeline, SA-WavLM considers the presence of all speakers in the mixture speech and models the interactions between them. In addition, a speaker shuffling strategy was introduced to ensure the model's invariance towards speaker order and absence. Experimental results showed that SA-WavLM has an enhanced ability to remove interference across various mixture speech tasks.

Table 3.4: Results on speech separation, including 1%, 10%, and 100% of training data (13,900 utterances for 100%).

Train Data	Pre-trained Model	Separation Model	SDRi↑	SI-SDRi↑
1%	–	ConvTasNet [Luo and Mesgarani, 2019]	3.05	2.59
	HuBERT Base [Hsu et al., 2021]		4.45	3.97
	WavLM Base [Chen et al., 2022a]		7.56	7.09
	SA-WavLM (Ours)		**8.81**	**8.42**
10%	–	ConvTasNet [Luo and Mesgarani, 2019]	9.73	9.16
	HuBERT Base [Hsu et al., 2021]		11.08	10.67
	WavLM Base [Chen et al., 2022a]		11.93	11.58
	SA-WavLM (Ours)		**13.93**	**13.62**
100%	–	ConvTasNet [Luo and Mesgarani, 2019]	14.53	14.12
	HuBERT Base [Hsu et al., 2021]		14.99	14.62
	WavLM Base [Chen et al., 2022a]		15.95	15.6
	SA-WavLM (Ours)		**16.55**	**16.22**

3.2 USED: Universal Speaker Extraction and Diarization

3.2.1 Introduction

A defining characteristic of speech communication is that each speaker carries distinct vocal traits. Within speech processing, this gives rise to two closely related front-end problems. Speaker extraction aims to recover the speech of a target speaker from a mixture by conditioning on that speaker's voiceprint [Zmolikova et al., 2023], whereas speaker diarization determines 'who spoke when' from the audio signal

[Park et al., 2022]. Both are foundational components in practical speech systems and are widely used in applications including speaker verification [Rao et al., 2019; Xu et al., 2021] and speech recognition [Žmolíková et al., 2017b; Delcroix et al., 2018; Radford et al., 2023; Mansfield et al., 2021].

From a broader perspective, these two problems are tightly connected. In multi-talker listening, the human brain effectively handles speaker extraction and diarization at the same time. By contrast, prior work has largely studied speaker extraction [Ge et al., 2020; Xu et al., 2020; Liu et al., 2023a] and diarization [Fujita et al., 2019; Horiguchi et al., 2020; Medennikov et al., 2020; Jiang et al., 2024; Chen et al., 2024d] as separate tasks. Such separation is limiting in realistic settings such as meeting analysis and mobile recording applications, where the output of only one task provides either speaking boundaries (timestamp information) or clean speech (content information), but not both. A more informative and interpretable result is obtained by linking extracted speech with speaker labels, thereby providing a complete account of 'who spoke what and when' [Chen et al., 2020b]. In addition, this association yields clean speech signals that can further support speaker-attributed automatic speech recognition, whose goal is to determine 'who spoke what' [Shafey et al., 2019; Kanda et al., 2020, 2021a].

This connection also suggests a natural complementarity between the two tasks, since both ultimately aim to disentangle individual speakers from a speech mixture. Speaker extraction can supply cleaner speech signals, which in turn can improve speaker activity detection in diarization. Conversely, diarization provides information about the presence or absence of the target speaker, and this information is helpful for speaker extraction.

Despite this complementarity, a joint treatment is nontrivial because the two tasks differ in several essential respects. The first difference lies in their output forms. A speaker extraction system [Ge et al., 2020; Xu et al., 2020; Liu et al., 2023a] typically

produces the voice of a single speaker, while a speaker diarization system [Fujita et al., 2019; Horiguchi et al., 2020; Medennikov et al., 2020] annotates speech activity for all speakers. The second difference lies in their typical operating conditions. Speaker extraction is usually developed for highly overlapped mixtures, for example when the speaker overlapping ratio is near 100% [Cosentino et al., 2020]. Speaker diarization, in contrast, mainly targets more sparsely overlapped conversational settings, where the overlap ratio is around 20% in meetings and daily conversations [Özgür Çetin and Shriberg, 2006; Barker et al., 2018]. In this work, these two obstacles are referred to as the 'output inconsistency' and 'scenario mismatch' issues, respectively, and they must be resolved for effective integration.

Existing studies have mostly addressed speaker extraction and speaker diarization either in isolation [Horiguchi et al., 2020; Medennikov et al., 2020; Ge et al., 2020; Liu et al., 2023a] or in joint formulations under controlled assumptions [Boeddeker et al., 2024; Maiti et al., 2023], such as a fixed number of speakers or a pre-defined overlap ratio. To date, no single-model approach has been proposed that performs both speaker extraction and speaker diarization for speech mixtures with varying overlap ratios and a variable number of speakers. This gap motivates the investigation of how the two tasks can be incorporated more effectively so that each can benefit from the other.

Motivated by this objective, this dissertation introduces the Universal Speaker Extraction and Diarization model (USED) to address both output inconsistency and scenario mismatch. The term 'universal' is used in two senses: first, the model can operate with a variable number of speakers; second, it can process speech mixtures with an arbitrary overlap ratio.

The model design is organized around two aspects, namely data ingress and data egress. On the ingress side, a novel embedding assignment module driven by speech references is introduced to mitigate output inconsistency in a natural manner. This

module offers two benefits: it enables the model to generate outputs for a variable number of speakers according to the provided speech references, thereby maintaining consistency across tasks; and it aligns the outputs of speaker extraction and speaker diarization so as to avoid the output permutation problem [Yu et al., 2017]. On the egress side, a multi-task interaction module is used to extend the effective overlap range and alleviate the scenario mismatch between the two tasks. Concretely, this module exploits diarization results to suppress background noise in extracted speech and to improve the optimization of the scenario-aware differentiated (SAD) loss [Pan et al., 2022] within USED. At the same time, speaker extraction provides cleaner speech signals that assist speaker diarization prediction, especially in overlapping regions.

The main contributions can therefore be summarized as follows. First, a universal joint solution, USED, is proposed for speaker diarization and speaker extraction, and it is designed to handle speech mixtures with an arbitrary number of speakers and diverse overlap ratios. Second, an embedding assignment module is introduced to maintain consistency in both the number and order of model outputs, which supports variable-speaker generation and improves robustness. Third, a multi-task interaction module is developed together with a scenario-aware differentiated loss, allowing the diarization output to determine whether the target speech should be silenced and thereby enforcing temporal overlap consistency between diarization and extraction outputs. Finally, the proposed USED model outperforms competitive speaker extraction and speaker diarization baselines on both the LibriMix and SparseLibriMix datasets. Its diarization capability is also evaluated on the CALLHOME dataset of real recordings, where the results show better performance with significantly less pre-training data.

3.2.2 Related Work

3.2.2.1 Speech Separation and Speaker Extraction

Recent studies on blind speech separation have made significant progress, with the development of advanced models such as Conv-TasNet [Luo and Mesgarani, 2019], Dual-Path Recurrent Neural Network (DPRNN) [Luo et al., 2020], Band-Split Recurrent Neural Network (BSRNN) [Luo and Yu, 2023] and efficient training algorithms such as Permutation Invariant Training (PIT) [Yu et al., 2017]. However, speech separation methods suffer from the global permutation ambiguity problem [Xu et al., 2020], which makes it hard to process long utterances. Moreover, it requires prior knowledge or estimation of the number of speakers in the speech mixture in advance, which is hard to get in real-world applications. Unlike blind speech separation that seeks to separate all speakers, speaker extraction only extracts speech for one target speaker at a time. In general, speaker extraction methods are classified into frequency-domain methods, such as SpeakerBeam [Žmolíková et al., 2017a; Delcroix et al., 2019; Žmolíková et al., 2019] and VoiceFilter [Wang et al., 2019], and time-domain methods, e.g. SpEx+ [Ge et al., 2020] and X-SepFormer [Liu et al., 2023a]. Time-domain approaches can naturally avoid the phase estimation problem, which exists in frequency-domain approaches [Ge et al., 2020]. Therefore, we develop the USED model based on the time-domain approach, SpEx+.

It is common that speech separation and speaker extraction systems in the literature are optimized for highly overlapped speech mixtures [Luo and Mesgarani, 2019; Ge et al., 2020; Liu et al., 2023a]. However, the overlap ratio in daily conversations and meetings is typically around 20% [Özgür Çetin and Shriberg, 2006; Barker et al., 2018], leading to a scenario mismatch between training on highly overlapped speech and evaluating on sparsely overlapped speech. To address such mismatch, speaker extraction for sparsely overlapped speech was recently studied [Borsdorf et al., 2021;

Zhang et al., 2020; Pan et al., 2022]. It was suggested to minimize speech power when the target speaker is absent, while maximizing the signal-to-distortion ratio (SDR) or scale-invariant signal-to-distortion ratio (SI-SDR) [Roux et al., 2019] when the target speaker is present. This introduces another challenge since it is hard to balance the two scenario-dependent objectives, i.e., SI-SDR and power, during training. We will study a multi-task interaction module in the USED model to overcome this challenge.

3.2.2.2 Speaker Diarization

Traditional clustering-based diarization methods [Shum et al., 2013; Sell and Garcia-Romero, 2014; Senoussaoui et al., 2014; Wang et al., 2018, 2024b] typically comprise multiple components that are trained separately. These methods implicitly assume that each speech segment is only from one speaker, which fails when multiple people speak at the same time. Two mainstream studies are attempting to solve this problem, i.e. end-to-end neural diarization (EEND) [Fujita et al., 2019; Horiguchi et al., 2020], and target speaker voice activity detection (TS-VAD) [Medennikov et al., 2020]. EEND seeks to minimize the diarization error directly with permutation-free objectives [Fujita et al., 2019], while TS-VAD uses the speaker embedding from the clustering results as the reference to detect the speaking status of each person. These methods have obtained good performance and set the stage for our study.

3.2.2.3 Interaction Between Speech Separation and Diarization

There were studies on the interaction between speech separation and diarization [Fujita et al., 2019; Wang et al., 2021b] by training speech processing models with multiple tasks. EEND-SS [Maiti et al., 2023] is a joint end-to-end framework for speaker diarization and separation. It utilizes the bottleneck feature of separation as one of the inputs for diarization and is optimized using multi-task learning. This model faces challenges related to permutation and processing very long sentences. In addition,

the fusion technique of EEND-SS is only applied during inference by using diarization probability directly. In TS-SEP [Boeddeker et al., 2024], it was proposed to pre-train a model with TS-VAD objective, and subsequently finetune the model to predict the mask for separation. In other words, TS-SEP is a two-stage framework that focuses on the speech separation task. Unlike EEND-SS and TS-SEP, our USED is a universal model that can concurrently handle both tasks and address their output inconsistency and scenario mismatch issues.

Two recent works [Kalda et al., 2024; Bando et al., 2024] have explored the joint training of speech separation and speaker diarization based on unsupervised speech separation approaches. PixIT [Kalda et al., 2024] employs PIT loss to integrate the speaker diarization task with MixIT [Wisdom et al., 2020] and draws on the *best-of-both-worlds* framework [Kinoshita et al., 2021; Kinoshita, Keisuke and Delcroix, Marc and Tawara, Naohiro, 2021] to stitch different segments based on the speaker diarization results. On the other hand, Neural FCASA [Bando et al., 2024] combines speaker diarization with neural FCA [Bando et al., 2021, 2022], eliminating the need for a separate speaker diarization step to mask source activities. Both approaches jointly address the two tasks under the premise of unsupervised speech separation, focusing on long-form audio and optimizing the final separation results in terms of diarization error rate (DER) and word error rate (WER). In comparison, our USED model emphasizes the performance of supervised speech separation and speaker diarization across different overlap ratios and scenarios.

3.2.3 Universal Speaker Extraction and Diarization

3.2.3.1 Task Definition

Let x be a multi-talker speech mixture with varying overlap ratio from 0% to 100 %, which consists of the speech signals from l speakers and a noise signal n. The speech

mixture can be represented as,

$$x = \sum_{i=1}^{l} c_i + n \tag{3.9}$$

where $l > 1$, and c_i denotes the speech signal of speaker i. Each speaker i has a corresponding speech reference, denoted as z_i, which is estimated from the speech mixture or provided as pre-enrolled information. Note that no normalization or noise reduction techniques are applied to the raw speech signals.

Given a speech mixture and speech references, the goal of the USED model is to generate separated speech signals and speaker activity information concurrently for a variable number of speakers, ranging from 1 to l. The number of speakers corresponds to the available speech references. This is motivated by scenarios where obtaining speech references for all speakers is challenging, or the target speakers are only a subset of all speakers. During inference, if there is only one target speaker, the USED model outputs a speech signal and diarization result for that speaker in a similar way to a speaker extraction model [Ge et al., 2020]. However, if there are l target speakers, the model would output multiple speech signals and a diarization results concurrently, one for each speaker, just like a speech separation model does [Luo and Mesgarani, 2019].

3.2.3.2 System Overview

As illustrated in Fig. 3.2, the USED model comprises six components: a speech encoder, a speaker encoder, an embedding processing module, a separator, an extraction decoder, and a diarization decoder.

The speaker encoder converts the available speech references into speaker embeddings, that are the feature representations of the target speakers. Meanwhile, the speech encoder generates spectrum or spectrum-like feature representations from the

speech mixture x.

Unlike TS-VAD [Medennikov et al., 2020], which generates diarization results for all speakers according to speaker embeddings and the speech mixture, we propose an embedding assignment module to enable custom control over the number of model's outputs. The embedding assignment module prepares the input embeddings with three different states, *active*, *blank*, and *residual*, based on the speaker embeddings, which are responsible for generating results accordingly. The *active* state is used as a position marker for the expected speakers. The *blank* state serves as a position marker to guide the network in outputting a silent segment. The *residual* state marks the residual unexpected speakers. The *blank* state is motivated by Connectionist Temporal Classification (CTC) [Graves et al., 2006], which integrates blank characters to manage pauses in speech recognition and gaps between characters in Optical Character Recognition (OCR). The embedding with *blank* state is designed to clearly distinguish the outputs from the expected and unexpected speakers. The reason for this design is that the output from an unexpected speaker is usually a silent segment. With the help of the embedding assignment module, the USED model naturally avoids the output inconsistency stemming from the integration of speaker extraction and diarization.

The separator then aims to separate speakers at the representation level from a speech mixture and embeddings from the embedding assignment module. Finally, diarization and extraction decoders predict the corresponding results based on the output of the separator.

The USED model differs from typical speaker extraction and speaker diarization systems [Fujita et al., 2019; Ge et al., 2020] in two key aspects. Firstly, it outputs speech signals for an arbitrary number of speakers, ranging from 1 to l, with the help of the embedding assignment module. Secondly, it seeks to effectively deal with speech mixtures with a wide range of overlapping ratios.

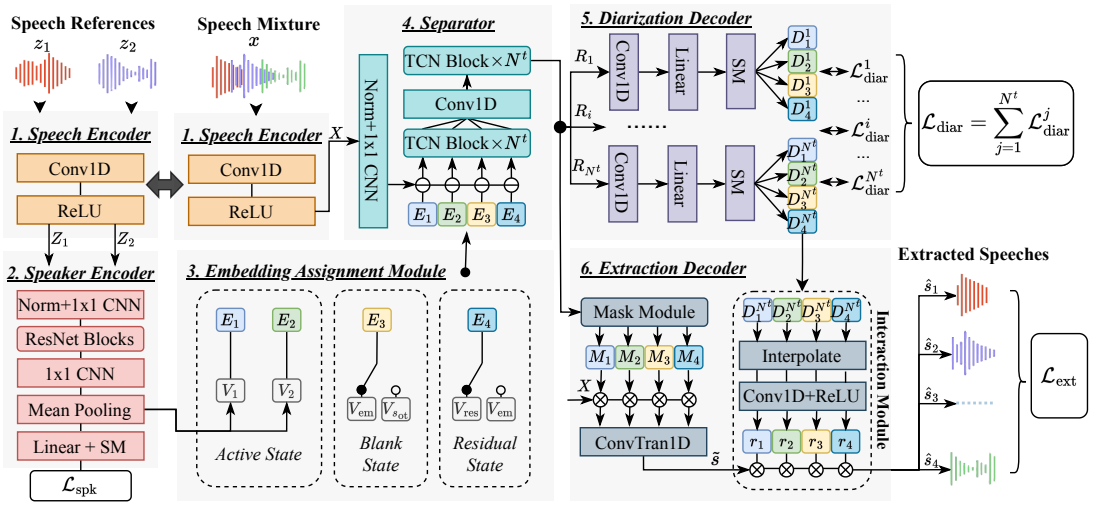


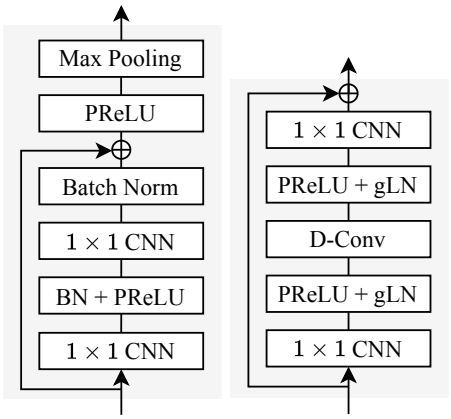
Figure 3.2: The overall training and inference framework of our proposed USED model. It comprises a speech encoder, a speaker encoder, an embedding assignment module, a separator, an extraction decoder and a diarization decoder. The USED model generates both the extracted speech and speaker diarization results by leveraging the speech mixture and the speech references as input. The symbols $\otimes$ and $\ominus$ refer to element-wise multiplication and frame-wise concatenation, respectively. The TCN block and mask module are denoted by rounded rectangles, which are described in detail in Fig. 3.3 and introduced later. Different colours are used to distinguish network layers from different components. The *active*, *blank* and *residual* states are assigned by Algorithm 1.

3.2.3.3 Speech Encoder

Analogous to a frequency analyzer, the speech encoder aims to generate spectrum-like frame-based embedding sequences from the speech mixture and speech references. Inspired by SpEx+[Ge et al., 2020], our approach leverages a multi-scale speech encoder.

Our multi-scale speech encoder combines three one-dimensional Convolutional Neural Networks (Conv1D), each with distinct kernel sizes to capture different scales. Subsequent to the convolutional layers, the activation function applied is the Rectified Linear Unit (ReLU).

The speech encoder produces spectrum-like frame-based embedding sequences



(a) ResNet Block (b) TCN Layer

Figure 3.3: Model structure for a ResNet block and a TCN layer. BN, PReLU, gLN and D-Conv are batch normalization, parametric ReLU, global layer normalization and dilated depth-wise separable convolution. The symbol $\oplus$ refers to element-wise addition.

X and Z_i for 3 scales as defined in [Ge et al., 2020] from the speech mixture x and the i-th speech reference z_i as

$$X = [X_1, X_2, X_3] = e(x) \in \mathbb{R}^{3\times T\times C} \tag{3.10}$$

$$Z_i = [Z_{i1}, Z_{i2}, Z_{i3}] = e(z_i) \in \mathbb{R}^{3\times T\times C} \tag{3.11}$$

where the function $e(\cdot)$ denotes the operation performed by the multi-scale speech encoder. The set $Z = \{Z_1, ..., Z_l\}$ represents the embedding sequences for l speakers. The input and output channel sizes of the three Conv1Ds are one and C, respectively. The kernel sizes of the three Conv1Ds are denoted as L_1^e, L_2^e and L_3^e, while they have a common stride of $\frac{L_1^e}{2}$. The outputs X and Z_i have a shape of $3 \times T \times C$, where T is the number of frames. For simplicity, Fig. 3.2 does not explicitly depict the multi-scale structure.

Algorithm 1: Assigning States for Embeddings

Data:
k: Maximum number of speakers the model can process
$S_{\text{mix\_spk}}$: Set of speaker IDs for the speech mixture
$S_{\text{all\_spk}}$: Set of speaker IDs for the whole dataset
V_{em}: Empty embedding
V_{res}: Residual embedding
p_{sel}: Probability for selecting present speaker
p_{em}: Probability to use empty embedding as input
p_{res}: Probability to use residual embedding as input

`// List of embeddings with active and blank state`
$I_{active}, I_{blank} \leftarrow [\,], [\,]$;
`// Number of embeddings`
$count \leftarrow 0$;
`/* The First Step for Active State */`
for $s_{mix} \in S_{mix\_spk}$ **do**
  if s_{mix} *is a present speaker* **then**
    $p \leftarrow$ random_uniform$(0, 1)$;
    `// If true, then marking embedding as` *`active`* `state`
    if $p > p_{sel}$ **then**
      $E_{count} \leftarrow V_{s_{\text{mix}}}$;
      I_{active}.add(E_{count});
      $count \leftarrow count + 1$;
    end
  end
end
`/* The Second Step for Blank State */`
while $count \leq k$ **do**
  $p \leftarrow$ random_uniform$(0, 1)$;
  if $p > p_{em}$ **then**
    `// Randomly select one speaker from the set of speakers that includes` $S_{\texttt{all\_spk}}$ `but excluding` $S_{\texttt{mix\_spk}}$
    $s_{\text{ot}} \leftarrow$random_sample$(S_{\text{all_spk}} \setminus S_{\text{mix_spk}})$;
    $E_{count} \leftarrow V_{s_{\text{ot}}}$;
  else
    $E_{count} \leftarrow V_{\text{em}}$;
  end
  I_{blank}.add(E_{count});
  $count \leftarrow count + 1$;
end
`/* The Third Step for Residual State */`
$p \leftarrow$ random_uniform$(0, 1)$;
if $p > p_{res}$ *or* I_{active}*.length* $\neq$ *number of present speakers* **then**
  $E_{k+1} \leftarrow V_{\text{res}}$;
else
  $E_{k+1} \leftarrow V_{\text{em}}$;
end
Output: $I_{active}, I_{blank}, E_{k+1}$

3.2.3.4 Speaker Encoder

The speaker encoder is designed to extract speaker embeddings from the corresponding enrolled speech. Specifically, we employ a speaker encoder to extract speaker embeddings, denoted as $V_1, ..., V_l$, from the embedding sequences Z. These speaker embeddings capture the distinctive characteristics of each speaker. After layer normalization (LN), we leverage a CNN layer with a kernel size of 1×1 (1×1 CNN) on the embedding sequence Z_i for speaker i. This is subsequently followed by N^r residual network (ResNet) blocks [He, Kaiming and Zhang, Xiangyu and Ren, Shaoqing and Sun, Jian, 2016]. The input and output channel sizes of this 1×1 CNN are $3C$ and C, respectively. Finally, an additional 1×1 CNN, combined with a mean pooling operation, generates a speaker embedding V_i for speaker i with an embedding dimension D_{spk}. After that, the speaker encoder predicts speaker identities, denoted as $\hat{y}_i^{\text{spk}}$, through a linear layer followed by a softmax (SM) activation function for speaker i.

The detailed architecture of the ResNet we used is shown in Fig. 3.3a. A ResNet block comprises two 1×1 CNNs and a max-pooling layer. Each 1×1 CNN within the ResNet block is accompanied by batch normalization (BN) and parametric ReLU (PReLU) for normalization and non-linear transformation. A skip connection adds the input to the output obtained after the second BN layer. Lastly, the max-pooling layer is responsible for adjusting the length of the output embedding.

We choose ResNet [He, Kaiming and Zhang, Xiangyu and Ren, Shaoqing and Sun, Jian, 2016] because it has become a prominent architecture in speaker verification systems [Chung et al., 2020; Zhou et al., 2021; Wang et al., 2023b; Zeinali et al., 2019] and is widely adopted in speaker extraction models [Ge et al., 2020; Wang et al., 2024c]. Its key advantage lies in the residual connections, which link frame-level layers and mitigate the vanishing gradient problem, facilitating faster convergence during backpropagation [He, Kaiming and Zhang, Xiangyu and Ren, Shaoqing and Sun, Jian, 2016]. These residual connections also enable deeper neural network con-

struction, allowing ResNet-based models to outperform regular CNNs. As a result, using ResNet as the backbone for speaker encoders leads to more robust and higher-quality speaker embeddings, making it a natural choice for this task.

3.2.3.5 Embedding Assignment Module

The embedding assignment module aims to prepare embeddings with different states for the separator. We set embeddings with different states: *active*, *blank*, and *residual*. The details of the embedding assignment module are shown in Algorithm 1.

Active State The embedding with *active* state is designed to guide the network to generate outputs for the expected speakers. Based on the embedding with *active* state, the whole speaker extraction and diarization process is driven by the speaker encoder that takes the speech references of some present speakers as input. Specifically, to enable our model to achieve the correct output for an arbitrary number of speakers, i.e., from 1 to l, we randomly assign the *active* state to the embeddings of present speakers by a hyperparameter p_{sel} representing probability, as shown in the first step of Algorithm 1.

Blank State The embeddings with *blank* state are primarily responsible for driving the network to produce silent segments for the unexpected speakers, which is critical in ensuring our network's robustness. Unlike *active* state, which requires a unique embedding from the speaker encoder, the embeddings with *blank* state alternate between two embeddings via a threshold p_{em} to guide the network: an inactive speaker embedding and a learnable empty embedding. The inactive speaker embedding is derived from a speaker who is not present in the speech mixture. This design ensures that our USED model can customize the number of outputs by inputting the learnable empty embedding during the inference stage.

Residual State As an additional state, the targets corresponding to the embedding with *residual* state are the cumulative outputs for speakers that are present but are not selected at the first step. This design allows our USED model to explore the interaction among all speakers in the speech mixture. Specifically, we also employ a learnable embedding V_{res} to teach the network to learn the corresponding cumulative outputs for the residuals. When speaker embeddings of all speakers have been considered as the *active* state, the target corresponding to *residual* state should be left without any speakers. In such cases, the choice between V_{res} and V_{em} is determined through a specified threshold p_{res}. Our preliminary experiments demonstrate that this strategy enhances robustness during the inference phase.

Overall, our USED model has $(k+1)$ input embeddings, where the first k embeddings with *active* and *blank* states support the predictions of at most k speakers and the $(k+1)$-th embedding E_{k+1} is with the *residual* state. Finally, we shuffle all the embeddings except E_{k+1} to guarantee the model's insensitivity to the order of embeddings. Through the designed embedding processing mechanism, our USED model can predict the outputs from a custom number of speakers. Additionally, the model's robustness is enhanced by employing a random selection strategy for speakers when assigning *blank* state.

3.2.3.6 Separator

The separator is designed to separate speakers based on the embeddings from the embedding assignment module. As shown in Fig. 3.2, the embedding sequence X firstly undergoes a series of operations, including a layer normalization followed by 1×1 CNN layer, wherein the input channel size is set to $3C$ and the output channel size is C. Then, we introduce N^t Temporal Convolutional Network (TCN) blocks [Luo and Mesgarani, 2019] to generate output representations based on the embeddings and speech mixture.

We use TCN as the backbone of our model because they have been widely adopted in speech separation and speaker extraction tasks[Luo and Mesgarani, 2019; Ge et al., 2020]. The USED model is built on the SpEx+ model, which also utilizes TCN. This choice ensures consistency and enhances performance. In addition, TCN is commonly used for time-domain approaches, avoiding the phase estimation issues found in time-frequency models. Finally, compared to methods like DPRNN [Luo et al., 2020] and BSRNN [Luo and Yu, 2023], Conv-TasNet [Luo and Mesgarani, 2019] offers lower computational complexity, reduced memory requirements, and faster training and inference.

Representations are then concatenated and down-sampled via a Conv1D layer with input channel size $(k+1) \times C$ and output channel size C. Finally, the processed representation is forwarded to another N^t TCN blocks to generate representations $\{R_1, ..., R_{N^t}\}$.

Each TCN block comprises a set of N^b TCN layers. The dilated depth-wise separable convolution (D-Conv) shown in Fig. 3.3b has an exponential growth dilation factor 2^b, where $b \in \{0, ..., N^b - 1\}$. In the initial TCN layer within each block, the input channel size for the 1×1 CNN is $D_{\text{spk}} + C$.

3.2.3.7 Diarization Decoder

The diarization decoder aims to predict the frame-level activity probabilities for each speaker from the N^t TCN blocks' outputs $\{R_1, ..., R_{N^t}\}$ in the separator. Because speaker extraction and diarization require different time resolutions, we employ a Conv1D layer to down-sample the outputs from the TCN block, with input and output channel sizes of C, a kernel size of L^{diar}, and a stride of $L^{\text{diar}}/2$. Subsequently, we use a linear layer followed by a softmax operation to compute the probabilities, denoted as $D^j = \{D^j_1, ..., D^j_{k+1}\}$, which represent the speech activities of the corresponding embeddings. Finally, we get a set of diarization results from each TCN block, denoted

as $\{D^1, ..., D^{N^t}\}$. During inference, we only use the diarization result D^{N^t} from the final TCN block as the results to calculate the evaluation metric.

3.2.3.8 Extraction Decoder

The extraction decoder is designed to estimate masks for each speaker and reconstruct the corresponding signals. The mask module of the extraction decoder comprises a Conv1D layer and ReLU activation to generate masks $\{M_1, ..., M_{k+1}\}$, where $M_i \in \mathbb{R}^{T \times C}$. We then obtain the modulated responses i by element-wise multiplication of the mask M_i and the representations X from the speech mixture.

For signal reconstruction, we employ a multi-scale decoder like speech encoder, which consists of three transposed convolutional networks (ConvTran1D), each with a kernel size of L_1^{e}, L_2^{e} and L_3^{e}, and a common stride, $L_1^{\mathrm{e}}/2$. It's used to reconstruct the time-domain signals, $\tilde{s} = \{\tilde{s}_1, ..., \tilde{s}_{k+1}\}$, where $\tilde{s}_i = \{\tilde{s}_i^1, \tilde{s}_i^2, \tilde{s}_i^3\}$ is obtained from three ConvTran1D layers.

As the USED model generates both the extracted speech and speaker diarization results, we propose an interaction module (IM) to use speaker diarization results to refine the generated waveforms. To avoid the gradient from the extraction task having an effect on the diarization task, we first create a clone of $D^{N^t} = \{D_1^{N^t}, ..., D_{(k+1)}^{N^t}\}$ without gradient. Subsequently, we employ interpolation techniques on the diarization output probabilities to ensure length-based uniformity between the two tasks. Following this alignment, a layer comprising a Conv1D layer and ReLU activation is employed to further process the output probabilities and obtain outputs r_i, ensuring that the values at each step are not constrained to be less than 1. The Conv1D layer possesses an input channel size of 1, an output channel size of 1, and a kernel size of L^{im}.

Finally, the extracted speech signals $\{\hat{s}_1, \ldots, \hat{s}_{k+1}\}$ are produced via element-wise

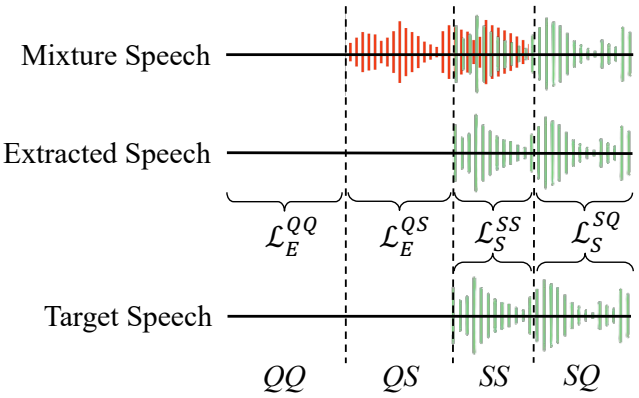


Figure 3.4: Illustration of scenario-aware differentiated loss. The speech mixture is segmented according to the four scenarios (*QQ*, *QS*, *SS* and *SQ*).

multiplication of $\{r_1, \ldots, r_{k+1}\}$ and the output $\tilde{s}$ from the extraction module as

$$\hat{s}_i = \{\hat{s}_i^1, \hat{s}_i^2, \hat{s}_i^3\} \tag{3.12}$$

where

$$\hat{s}_i^j = \tilde{s}_i^j \otimes r_i \tag{3.13}$$

It should be noted that, during inference, only $\tilde{s}_1^1, ..., \tilde{s}_k^1$ originating from the first ConvTran1D are utilized for the evaluation of metrics.

3.2.3.9 Loss Function

Overall Loss Function The loss of the USED model can be formulated as follows:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{\text{ext}} + \lambda_2 \mathcal{L}_{\text{diar}} + \lambda_3 \mathcal{L}_{\text{spk}} \tag{3.14}$$

where λ_1, λ_2 and λ_3 are the weights for speaker extraction loss, speaker diarization loss, and speaker classification loss, respectively.

Speaker Extraction We adopt the SAD loss proposed in [Pan et al., 2022] to address sparsely overlapped speech as shown in Fig. 3.4. Scenarios are classified into four distinct classes: *QQ*, *QS*, *SS*, and *SQ*. In the *QQ* scenario, both the target speaker and interference speakers are in a *quiet* state. The *QS* scenario represents a situation where the target speaker is *quiet* while the interference speakers are *speaking*. Conversely, in the *SQ* scenario, the target speaker is *speaking* while the interference speakers are *quiet*. Lastly, the *SS* scenario corresponds to both the target speaker and interference speakers being in a *speaking* state. For *QS* and *QQ*, we use power as the objective for extracted speech $\hat{s}$,

$$\mathcal{L}_{\text{E}} = \sum_{j=1}^{3} 10\mu_j \log_{10}(\frac{\|\hat{s}^j\|^2}{T_{\text{se}}} + \epsilon) \tag{3.15}$$

where T_{se} is the duration of $\hat{s}^j$ in seconds, $[\mu_1, \mu_2, \mu_3]$ are the weights for different ConvTran1D layers and ϵ is set to 10^{-6}. For *SS* and *SQ*, we compute the SI-SDR loss between the target speech s and extracted speech $\hat{s}$ as $\mathcal{L}_{\text{S}}$,

$$\mathcal{L}_{\text{S}} = \sum_{j=1}^{3} -10\mu_j \log_{10} \frac{\|\frac{<\hat{s}^j, s>s}{\|s\|^2+\epsilon}\|^2}{\|\hat{s}^j - \frac{<\hat{s}^j, s>s}{\|s\|^2+\epsilon}\|^2 + \epsilon} + \epsilon \tag{3.16}$$

SAD loss is used to calculate the loss between the extracted speeches $\{\hat{s}_1, .., \hat{s}_{k+1}\}$ generated from the multi-task interaction module and target speeches $\{s_1, ..., s_{k+1}\}$. For clarity, we assume the first m target speeches correspond to the embeddings with *active* state and are the same as $\{c_1, ..., c_m\}$ in Eq. (3.9). $\{s_{m+1}, ..., s_k\}$ are zero vectors which present silence as the target of embeddings with *blank* state. s_{k+1} is the target speech for residual prediction. In practice, the order is shuffled as introduced in Section 3.2.3.5. Each pair of extracted speech and target speech are firstly segmented according to the four scenarios. Finally, the total speaker extraction loss, i.e. $\mathcal{L}_{\text{ext}}$, is as follows:

$$\mathcal{L}_{\text{ext}} = \frac{1}{k+1}(\sum_{i=1}^{k+1} \alpha \mathcal{L}_{\text{E}_i}^{QQ} + \beta \mathcal{L}_{\text{E}_i}^{QS} + \gamma \mathcal{L}_{\text{S}_i}^{SS} + \delta \mathcal{L}_{\text{S}_i}^{SQ}) \tag{3.17}$$

where $k+1$ is the total number of outputs and α, β, γ and δ are the weights for different scenarios.

Speaker Diarization To optimize the model for the diarization task, we apply the binary cross-entropy loss (BCE), denoted as $\mathcal{L}_{\text{diar}}^{j}$ on the output of the N^t TCN blocks,

$$\mathcal{L}_{\text{diar}}^{j} = \frac{1}{k+1} \sum_{i=1}^{k+1} BCE(y_i^{\text{diar}}, D_i^j) \tag{3.18}$$

where y_i^{diar} is the ground-truth label for the i-th output. Then, the entire speaker diarization loss can be represented as:

$$\mathcal{L}_{\text{diar}} = \sum_{j=1}^{N^t} \mathcal{L}_{\text{diar}}^{j} \tag{3.19}$$

Speaker Classification The speaker encoder is optimized through a cross-entropy loss (CE), denoted as $\mathcal{L}_{\text{spk}}$, for speaker classification during the training process, as

$$\mathcal{L}_{\text{spk}} = \frac{1}{l} \sum_{i=1}^{l} CE(y_i^{\text{spk}}, \hat{y}_i^{\text{spk}}) \tag{3.20}$$

where y_i^{spk} and $\hat{y}_i^{\text{spk}}$ are the ground truth and predicted speaker ids for speaker i, respectively.

3.2.4 Experimental Setup

3.2.4.1 Datasets

We conduct comprehensive evaluations of our model using diverse datasets, including LibriMix and SparseLibriMix, which are simulated, and CALLHOME, which consists of real recordings.

LibriMix The first dataset, known as LibriMix [Cosentino et al., 2020], generates samples derived from LibriSpeech [Panayotov et al., 2015a] train-clean-100 and test-clean datasets, as well as WHAM! [Wichern et al., 2019], all with a 16kHz sampling rate. LibriMix has been widely used for speech separation [Li et al., 2023; wen Yang et al., 2021], speaker extraction [Ge et al., 2022] and speaker diarization tasks[wen Yang et al., 2021; Maiti et al., 2023]. We merge the Libri2Mix 100h and Libri3Mix 100h datasets, resulting in utterances that may contain either 2 or 3 speakers.

The models are evaluated on both min and max modes. The min mode consists of highly overlapped speech segments, while a larger portion of the max mode consists of non-overlapped speech segments. To ensure compatibility with the speaker extraction task, we have slightly adjusted the data split for training and validation, as previously done in related work [Ge et al., 2022].

SparseLibriMix The second dataset employed in our evaluation is the test set from SparseLibriMix [Cosentino et al., 2020], which utilizes WHAM! [Wichern et al., 2019] to simulate a noisy acoustic environment. This dataset covers more realistic mixture scenarios, varying from an overlap ratio of 0 to 1.0. It contains 1,000 utterances which have 2 or 3 speakers.

CALLHOME We use the CALLHOME dataset [Mark and Martin, 2001] to evaluate the diarization task with real recordings. The CALLHOME dataset is partitioned into

two parts based on the Kaldi recipe. The first part, known as Part 1, is utilized for model adaptation, whereas the second part, referred to as Part 2, is used for evaluation. The CALLHOME dataset consists of telephone-channel recordings with 8k sample rate. Besides CALLHOME Part 1, we further add Libri2Mix 100h, Libri3Mix 100h, Libri2Mix 360h and Libri3Mix 360h, which are generated from LibriSpeech [Panayotov et al., 2015a] train-clean-100 and train-clean-360, as the pre-training dataset. In total, our dataset encompasses approximately 452 hours of data. We then finetune the model only with CALLHOME Part 1 and evaluate on CALLHOME Part 2. During inference, we follow the evaluation pipeline of TS-VAD to evaluate our model so that it does not require pre-enrolled speech. Specifically, we extract the speech references based on the diarization result of spectral clustering and then do inference for USED models. The speaker embedding for spectral clustering is extracted from the ECAPA-TDNN model [Desplanques et al., 2020] trained on VoxCeleb2 [Chung et al., 2018].

3.2.4.2 Baseline Configuration

Two of the baseline models, SpEx+ [Ge et al., 2020], and TS-VAD [Medennikov et al., 2020], are implemented by ourselves. During the training phase, we segment the utterances into chunks, each of which spans 4 seconds and is shifted every 2 seconds.

For the SpEx+ baseline, we use the same training configuration as the USED model described in Section 3.2.4.3. To align the model complexity with that of the USED model, we configure the SpEx+ baseline with a total of 8 TCN blocks so that it has a similar number of parameters to the USED model. It is worth noting that the SpEx+ baseline can be regarded as a specialized version of the USED model, as it has only one embedding with *active* state as input without including a diarization module and SAD loss. Our metric for selecting the best checkpoint for the SpEx+ model is SI-SDR.

As for the TS-VAD baseline, we follow [Medennikov et al., 2020] to design the

model, with several notable modifications. Firstly, we substitute the bidirectional LSTM in the original TS-VAD model with four transformer layers, with sine-cosine positional encoding as input. The model architecture entails two layers dedicated to processing inputs for each speaker individually, followed by a Conv1D layer for down-sampling. Another two layers are used to combine the information from all speakers. For the transformer, we set the embedding dimension to 384 and employ four attention heads. Second, we use a pre-trained speaker encoder, ECAPA-TDNN [Desplanques et al., 2020], to extract speaker embeddings. This speaker encoder also takes on the role of replacing the CNN encoder to extract time-level representations from the speech mixture. Finally, we use a similar strategy as stated in Section 3.2.3.5, which randomly selects speaker embeddings from the whole dataset. p_{em} is set to be 0.3. The model is optimized using Adam with batch size 64 for 40k steps. The speech encoder is kept fixed for the first 4,000 steps, and the learning rate follows a polynomial decay schedule with an initial value of 2×10^{-4}, which has a warm-up phase for the first 10% of steps. The DER metric is our criterion for selecting the best checkpoint for the TS-VAD model.

3.2.4.3 USED Model Configuration

The maximum number of speakers k is 3 for evaluation on the LibriMix and SparseLibriMix datasets and in the assessment on the CALLHOME dataset. For the multi-scale speech encoder, C is 256 and the kernel sizes, L_1^e, L_2^e and L_3^e, are 20, 80 and 160. The number of ResNet blocks N^r in the speaker encoder is set to 4, and the dimension of the speaker embedding D_{spk} is 256. Regarding the separator module, we set N^t to 3 and N^b to 8. For the diarization decoder, the CNN layer has a kernel size L^{diar} of 32 and stride $L^{\text{diar}}/2$ of 16. The mask module of the extraction decoder consists of a CNN with a kernel size of 1 and stride 1, and ReLU activation. For the multi-task interaction module, the kernel size L^{im} of the CNN layer is 16. The loss weights for

outputs from different ConvTran1D layers, μ_1, μ_2 and μ_3 are 0.8, 0.1 and 0.1 by following [Ge et al., 2020]. We set 0.001 for α and β, and 1.0 for γ and δ, which are the loss weights in Eq. (3.17). λ_1, λ_2, and λ_3 are set to 1.0. p_{em} is set to 0.3. p_{sel} and p_{res} are set to 0.5 and 0.9, respectively, when *residual* state is applied. The USED model without *residual* state and with $p_{sel} = 0$ is called as "USED-F(ix)" since it always gets speech references of present speakers during training.

We optimize the model with Adam on 4 GPUs for 100k steps, corresponding to approximately 11 epochs. The utterances are split into chunks with a size of 4 seconds and a shift of 2 seconds. We set the maximum tokens to 260k, which results in processing around 16 seconds of audio for each batch. The learning rate is set to $1e-3$ for all experiments, which is warmed up for the first 10% steps and decayed polynomially for the remaining steps. The overall loss value is our primary metric for selecting the best checkpoint for the USED model.

3.2.4.4 Evaluation Metrics

We use diarization error rate (DER (%)), including speaker confusion (SC (%)), false alarm (FA (%)), and missed detection (MS (%)) to evaluate the speaker diarization performance. Collar tolerance and median filtering are set to 0 seconds and 11 frames for the LibriMix and SparseLibriMix datasets, and 0.25 seconds and 11 frames for the CALLHOME datasets. We report SI-SDR improvement (SI-SDRi (dB)), Power (dB/s), SDR improvement (SDRi (dB)), Short-Time Objective Intelligibility (STOI) [Taal et al., 2010], and Perceptual Evaluation of Speech Quality (PESQ) [Rix et al., 2001] for speaker extraction. For complexity analysis, we use MultiplyAccumulate Operations (MACs (G)) to assess computational cost and report the model's parameter count.

3.2.5 Experimental Results and Analysis

3.2.5.1 Results on the LibriMix Dataset

Tables 3.5 and 3.6 show the results on the LibriMix dataset with max mode and min mode, respectively. As indicated before, min mode mainly contains highly overlapped speech, while a larger proportion of max mode is non-overlapped speech compared to min mode. We provide detailed results under different scenarios for the max mode as shown in Table 3.5. Please note that we use the average duration length that anyone is speaking in seconds as the metric of speaker diarization for the *QQ* scenario.

Our analysis involves comparisons with other popular systems from the literature, including TS-VAD [Medennikov et al., 2020], EEND-EDA [Horiguchi et al., 2020], HuBERT Base [Hsu et al., 2021] and wav2vec 2.0 Base [Baevski et al., 2020b] for speaker diarization, SpEx+ [Ge et al., 2020], and Conv-TasNet [Luo and Mesgarani, 2019] for speaker extraction and speech separation, and EEND-SS [Maiti et al., 2023] for joint optimization. For HuBERT Base and wav2vec 2.0 Base, we firstly follow the SUPERB diarization task [wen Yang et al., 2021] to train two downstream models for two speakers and three speakers, respectively. Subsequently, we combine the output results of these two models for evaluation.

The proposed USED model demonstrates superior or comparable performance compared to the baseline systems, as indicated by their respective evaluation metrics in both the min and max modes. For speaker diarization, our models exhibit remarkable improvements, with a relative reduction in DER of at least 13% when compared to the prior systems. For speaker extraction, our models achieve a much lower value in terms of power compared to other baseline systems under QQ and QS scenarios in the max mode. This phenomenon implies that speaker diarization and SAD loss effectively mitigate background noise during silent intervals, resulting in enhanced performance.

Table 3.5: Performance on the LibriMix dataset for **max mode**. The results of baseline systems are obtained by our own implementations.

Speaker Diarization							
Model	Overall Performance				SS	SQ&QS	QQ
	DER↓	MS↓	FA↓	SC↓	DER↓	DER↓	Seconds↓
wav2vec 2.0 Base [Baevski et al., 2020b]	7.62	2.28	4.82	0.52	-	-	-
HuBERT Base [Hsu et al., 2021]	7.56	2.40	4.81	**0.35**	-	-	-
TS-VAD [Medennikov et al., 2020]	7.28	3.61	2.78	0.89	5.71	11.63	0.08
USED-F	**4.75**	**2.18**	2.16	0.42	**3.21**	**9.41**	0.05
USED	5.20	2.68	**2.08**	0.43	3.56	10.36	**0.04**
Speaker Extraction							
Model	Overall Performance				SS	SQ	QS&QQ
	SI-SDRi↑	SDRi↑	STOI↑	PESQ↑	SI-SDRi↑	SI-SDRi↑	Power↓
SpEx+ [Ge et al., 2020]	9.06	10.11	0.776	1.39	8.32	5.17	55.60
USED-F	**12.70**	**13.22**	**0.844**	**1.46**	**12.00**	9.17	**-24.00**
USED	12.22	12.69	0.836	1.43	11.63	**9.30**	-14.15

3.2.5.2 Results on SparseLibriMix

In addition to the LibriMix dataset, we are also interested in assessing the performance of our model across a range of overlap ratios. Therefore, we conduct an evaluation on SparseLibriMix, as illustrated in Fig. 3.5. We select TS-VAD and SpEx+ models as the baseline systems. The model trained on the max mode of Libri2Mix and Libri3Mix is evaluated on SparseLibriMix with 2 and 3 speakers. The proposed USED models outperform each task baseline, confirming our models' effectiveness for variable values of overlap ratio.

3.2.5.3 Results on CALLHOME

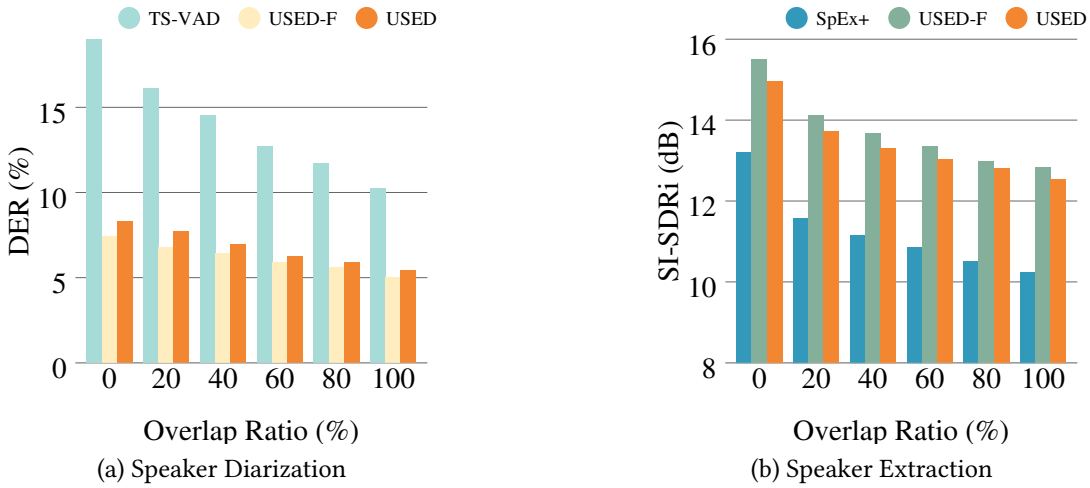
To compare the performance of diarization on real data, we evaluate our model on the CALLHOME dataset. The results are summarized in Table 3.7, compared with several speaker diarization methods, including two clustering-based methods, AHC clustering [Horiguchi et al., 2020] and spectral clustering [Wang et al., 2018], PixIT [Kalda et al., 2024], end-to-end speaker diarization models, EEND-EDA [Horiguchi

Table 3.6: Performance on the LibriMix dataset for **min mode**. † indicates the results are obtained by our own implementations.

Speaker Diarization				
Model	DER ↓	MS ↓	FA ↓	SC ↓
EEND-EDA [Maiti et al., 2023]	10.16	-	-	-
TS-VAD †[Medennikov et al., 2020]	5.30	2.32	2.83	0.15
EEND-SS [Maiti et al., 2023]	6.27	-	-	-
+ LMF	6.04	-	-	-
USED-F	4.64	1.99	2.55	0.10
USED	**4.57**	**1.94**	**2.54**	**0.09**
Speaker Extraction / Speech Separation				
Model	SI-SDRi ↑	SDRi ↑	STOI ↑	PESQ ↑
Conv-TasNet [Maiti et al., 2023]	7.66	8.71	0.756	-
SpEx+ †[Ge et al., 2020]	8.76	10.09	0.772	1.44
EEND-SS [Maiti et al., 2023]	9.31	7.50	0.760	-
+ Fusion	9.38	7.59	0.760	-
+ LMF	8.83	9.72	0.767	-
+ LMF + Fusion	8.87	9.77	0.767	-
USED-F	**12.24**	**12.83**	**0.841**	**1.54**
USED	11.98	12.50	0.834	1.44

et al., 2020], SC-EEND [Fujita et al., 2020] and AED-EEND [Chen et al., 2023], and the TS-VAD model [Medennikov et al., 2020]. Most of those methods use the simulation data from Swichboard-2, Switchboard Cellular, and NIST Speaker Recognition Evaluation [Horiguchi et al., 2020], which have a variable number of speakers, ranging from one to five. The total number of hours for the pre-training data is around 15,516 hours for those methods. In contrast, we use much less simulation data, which is from LibriMix and only have 2 or 3 speakers. This results in around 452 hours of pre-training data.

Our USED model achieves around 15% relative improvement on overall DER compared to the TS-VAD baseline and performs better with much less pre-training data compared to other methods. We observe that our USED model is more robust in terms



(a) Speaker Diarization (b) Speaker Extraction

Figure 3.5: Performance on the SparseLibriMix for overlap ratio from 0% to 100%. The Left and right bar charts are the results of speaker diarization and speaker extraction tasks, respectively.

of the number of speakers. Results show that although our model only uses simulation data from LibriMix, which has a speech mixture with 2 or 3 speakers, the model still achieves better DER results when the number of speakers is larger than 3. We believe this gain is from the design of the embedding assignment module.

3.2.5.4 Investigation of USED Models for Generating Results of Different Numbers of Speakers During Inference

In this section, we compare the USED-F model and the USED model with baseline systems when generating results for different numbers of speakers during inference, as shown in Table 3.8. As introduced in Section 3.2.3.9, m is the number of embeddings with an *active* state as the input of the USED model. During inference, m also represents the number of speakers the USED model is required to predict. When m is 1, USED models perform speaker extraction like SpEx+. Under this condition, USED models and SpEx+ have the same input data for a fair comparison. When m is all, it means we provide all speech references for present speakers, and the embedding is

Table 3.7: DER (%) results on CALLHOME, a dataset of real recordings. The symbol ‡ denotes results obtained from our own implementations, while † indicates results evaluated using the open-source model.

Model	Pre-training Data	Number of Speakers					
		2	3	4	5	6	all
AHC clustering [Horiguchi et al., 2020]	-	15.45	18.01	22.68	31.40	34.27	19.43
spectral clustering [Wang et al., 2018]	-	16.05	18.77	22.05	30.46	36.85	19.83
PixIT †[Kalda et al., 2024]	79h	17.02	26.03	27.75	36.38	40.77	24.41
EEND-EDA [Horiguchi et al., 2020]	15,516h	8.50	13.24	21.46	33.16	40.29	15.29
SC-EEND [Fujita et al., 2020]	15,516h	9.57	14.00	21.14	31.07	37.06	15.75
AED-EEND [Chen et al., 2023]	15,516h	**6.96**	**12.56**	18.26	34.32	44.52	14.22
TS-VAD ‡ [Medennikov et al., 2020]	452h	12.95	14.57	20.26	27.36	32.66	15.51
USED-F	452h	9.09	12.76	**14.67**	**26.96**	26.71	**13.16**
USED	452h	10.23	12.78	14.68	32.02	**25.10**	13.51

V_{em} for *residual* state. Otherwise, we use V_{res} as the embedding for *residual* state.

First, USED models consistently perform much better than the SpEx+ baseline when m is 1, except for the USED-F model. This indicates that even if the USED model only gets one speech reference, it can still outperform the SpEx+ model. When m is 2 or all, the performance of USED in speaker extraction significantly surpasses that of SpEx+. The experimental results show that the USED model demonstrates superior efficiency and performance in speaker extraction tasks, particularly in multi-speaker scenarios. Unlike SpEx+, which requires multiple inferences for each speaker, USED performs inference just once, reducing redundant computations like those in the speech encoder. It can also simultaneously utilize the embeddings of multiple speakers, resulting in more discriminative results, whereas SpEx+ processes one speaker at a time. As the number of speakers increases, USED's performance advantage becomes even more significant.

Second, when extracting results for all speakers, the USED model outperforms TS-VAD, as shown in the experimental results. In addition, the TS-VAD model can only extract results for all speakers simultaneously, while the USED model provides flexibility to extract results for a specific speaker or multiple speakers, which is ad-

vantageous in scenarios where obtaining embeddings for all speakers at once is not feasible. Furthermore, the USED model does not require a pre-trained speaker encoder, unlike TS-VAD, which typically relies on one.

Third, we observe that the USED model without *residual* state achieves improvement compared to the USED-F model on both speaker diarization and extraction tasks if m equals 1 or 2. It gets a larger improvement when applying *residual* state with p_{res} of 1.0. However, when m is all, the performance of these two models degrades significantly on the speaker diarization task compared to the USED-F model with the DER increasing from 4.75% to 10.38% and 9.46%, respectively. This degradation can be avoided by setting p_{res} to 0.9 during training. In this way, the USED model can achieve a competitive performance compared to that of the USED-F model if m is all and still significantly outperforms baseline systems when m is 1 or 2.

The performance gap between USED-F and USED models when $m =$ all is due to the fact that the input to the USED-F model consistently includes speaker embeddings for all speakers during training. As a result, USED-F is designed to predict for all speakers simultaneously, making it specifically optimized for scenarios where $m =$ all. However, its performance is less robust in cases where $m \neq$ all (e.g., $m = 1$ or $m = 2$), leading to lower performance compared to the USED model in these situations.

3.2.5.5 Ablation Study: Assessing Single-Task Training for the USED Model

We then compare USED models trained on single task to investigate the impact of each task as illustrated in Table 3.9. Without loss of generality, we use the USED-F model for comparison. We present the results obtained by setting λ_1 or λ_2 to 0, referred to as Only SD (Speaker Diarization) and Only SE (Speaker Extraction), respectively.

The USED-F model achieves a notable reduction of DER with a relative improve-

Table 3.8: Model performance on the LibriMix dataset for max mode when setting different values of m. RS stands for residual state.

m	Model	SI-SDRi ↑	DER ↓
1	SpEx+ [Ge et al., 2020]	9.06	-
	USED-F	5.71	20.30
	USED	9.73	11.13
	- w/o RS	9.87	11.96
	- $p_{\text{res}} = 1.0$	**10.23**	**10.85**
2	USED-F	9.55	11.27
	USED	11.02	**9.42**
	- w/o RS	10.96	10.98
	- $p_{\text{res}} = 1.0$	**11.08**	10.14
all	TS-VAD [Medennikov et al., 2020]	-	7.28
	USED-F	**12.70**	**4.75**
	USED	12.22	5.20
	- w/o RS	11.80	9.46
	- $p_{\text{res}} = 1.0$	11.76	10.38

ment of around 32% in the SS scenario and 20% in the SQ&QS scenarios. The larger improvement in the SS scenario suggests that speaker extraction indeed plays a pivotal role in enhancing the diarization module's ability to handle overlapping speech segments. Furthermore, SI-SDRi increases from 12.46 dB to 12.70 dB for overall performance when compared to the USED-F model with only SE, which may be mainly from the improvement under the QQ scenario where the power reduces from 20.84 dB/s to -24.00 dB/s. This indicates that the speaker diarization task helps surpass the background noise for the speaker extraction task.

3.2.5.6 Investigation of the Relationship between the Multi-Task Interaction Module and the SAD Loss

In this section, we analyze the performance of the USED-F model using different weights for each scenario of SAD loss with or without the multi-task interaction module as illustrated in Table 3.10. For the experiments without the multi-task interaction

Table 3.9: Ablation study of the USED model configured for a single task (Only SD or Only SE) on the LibriMix dataset for max mode.

Speaker Diarization				
Model	All DER↓	SS DER↓	SQ&QS DER↓	QQ Seconds↓
USED-F	4.75	3.21	9.41	0.05
- Only SD	6.43	4.72	11.76	0.04
Speaker Extraction				
Model	All SI-SDRi↑	SS SI-SDRi↑	SQ SI-SDRi↑	QS&QQ Power↓
USED-F	12.70	12.00	9.17	-24.00
- Only SE	12.46	11.71	9.03	20.84

module, results demonstrate that when increasing the weights α and β for *QQ* and *QS* scenarios, we can get a lower power value, but the SI-SDRi are reduced from 12.02 dB to 10.48 dB for the *SS* scenario and from 9.58 dB to 7.42 dB for the *SQ* scenario. Regarding the performance of speaker diarization, DER results increase when larger values for the weights α and β of *QQ* and *QS* scenarios are used. These observations indicate that the model performance is sensitive to the loss weights under different scenarios. With the help of the multi-task interaction module, we can keep the performance for speaker extraction under *SS* and *SQ* scenarios and speaker diarization and simultaneously achieve a much lower power under the QQ scenario, which decreases from -1.93 dB/s to -24.00 dB/s.

3.2.5.7 Complexity Analysis

In this section, we conduct a complexity analysis of the USED model in comparison to the baseline models, TS-VAD and SpEx+, as shown in Table 3.11. To ensure a fair comparison, the duration of the speech input for all models is kept consistent, with both the speech mixture and the speech references set to 4 seconds. The experimental

Table 3.10: USED-F model performance under different scenarios when using different loss weights for each scenario of SAD loss with or without Multi-Task interaction module. IM means multi-task interaction module. The results of the first line are from the standard setting.

α	β	γ	δ	IM	**Speaker Extraction** SI-SDRi ↑ *All*	SI-SDRi ↑ *SS*	SI-SDRi ↑ *SQ*	Power ↓ *QS & QQ*	**Speaker Diarization** DER↓
0.001	0.001	1.0	1.0	✓	12.70	12.00	9.17	-24.00	4.75
0	0	1.0	1.0	✗	12.47	12.02	9.58	55.68	4.80
0.001	0.001	1.0	1.0	✗	12.71	12.00	9.11	-1.93	4.65
0.01	0.01	1.0	1.0	✗	12.50	11.65	8.68	-45.16	4.77
1.0	1.0	1.0	1.0	✗	11.54	10.48	7.42	-79.21	5.57

Table 3.11: Model Comparison in Terms of Parameters and MACs for the TS-VAD model, the SpEx+ model and the USED model. "Params" refers to the number of model parameters, which is counted in millions (M). †The MACs of SpEx+ is only for extracting one speaker's result.

Model	Params	MACs	Total MACs
TS-VAD [Medennikov et al., 2020]	39.50	27.21	158.01
SpEx+ †[Ge et al., 2020]	16.35	43.60	
USED-F	23.12	96.91	96.91
USED	23.65	119.54	119.54

results indicate that, compared to individual baseline models, the USED model has more parameters than the SpEx+ model but fewer than the TS-VAD model. However, the computational complexity of the USED model is higher than both SpEx+ and TS-VAD.

It is important to note that the USED model simultaneously addresses two tasks: speaker extraction and diarization, whereas TS-VAD and SpEx+ are designed for a single task, with SpEx+ specifically extracting the speech for only one speaker. Therefore, we calculate the 'Total MACs'. 'Total MACs' refers to the total number of MACs required for the models to process a speech mixture containing three speakers and ob-

Table 3.12: Model performance regarding different loss weight values on the validation set.

λ_1	λ_2	$\lambda_3 = 0.1$		$\lambda_3 = 0.5$		$\lambda_3 = 1.0$	
		SI-SDRi ↑	DER ↓	SI-SDRi ↑	DER ↓	SI-SDRi ↑	DER ↓
0.1	0.1	14.31	4.15	14.27	4.26	14.27	4.15
0.1	0.5	14.01	3.85	13.99	3.76	14.06	3.74
0.1	1.0	13.54	3.85	13.46	4.00	13.32	4.19
0.5	0.1	14.22	5.01	14.13	5.14	14.19	5.01
0.5	0.5	14.16	4.23	14.27	4.26	14.32	4.14
0.5	1.0	14.19	3.97	14.32	3.92	14.28	3.84
1.0	0.1	14.11	5.49	14.15	5.37	14.17	5.47
1.0	0.5	14.15	4.57	14.18	4.55	14.32	4.45
1.0	1.0	14.22	4.21	14.32	4.14	14.29	4.22

tain the corresponding speaker extraction and diarization results for all three speakers. When comparing the 'Total MACs', the 'Total MACs' of the USED and USED-F models are significantly lower than that of the baseline models combined. Specifically, the USED-F model reduces the total MACs by approximately 39%, while the USED model achieves a reduction of around 24%.

3.2.5.8 Loss Weight Selection

To investigate the impact of different loss weights on the model's performance and explore how to select appropriate values for these weights, we conduct a series of experiments with three loss weights (λ_1, λ_2, and λ_3), as illustrated in Table 3.12. We evaluate the model's performance on the validation set of the LibriMix dataset with max mode by testing each loss weight with values of 0.1, 0.5, and 1.0. The experimental results indicate that there may be room for improvement in the model's performance regarding the selection of loss weights, although we set the values of all three weights to 1.0 for other experiments. For example, when λ_1, λ_2, and λ_3 are 0.5, 1.0 and 0.5, respectively, the USED model performs slightly better on both the speaker extraction and diarization tasks.

Table 3.13: Impact of hyperparameter choices for embedding assignment module on model performance.

m	Metrics	p_{sel}					p_{em}					p_{res}				
		0.1	0.3	0.5	0.7	0.9	0.1	0.3	0.5	0.7	0.9	0.1	0.3	0.5	0.7	0.9
1	SI-SDRi ↑	11.81	10.36	12.12	12.81	12.60	10.77	12.12	12.93	13.09	13.10	13.12	12.99	12.72	13.01	12.12
	DER ↓	6.80	8.79	6.47	5.83	6.19	6.67	6.47	5.87	5.63	5.62	5.75	5.83	6.21	5.76	6.47
2	SI-SDRi ↑	13.28	13.30	13.58	13.62	12.81	13.68	13.58	13.83	13.56	13.42	13.77	13.76	13.83	13.72	13.58
	DER ↓	5.17	5.11	4.98	5.28	9.41	5.01	4.98	5.32	4.97	5.05	5.67	5.15	5.24	4.93	4.98
all	SI-SDRi ↑	14.71	14.45	14.29	14.07	13.10	14.25	14.29	14.31	14.26	14.27	14.31	14.28	14.35	14.25	14.29
	DER ↓	3.87	4.00	4.22	4.41	6.27	4.30	4.22	4.16	4.13	4.18	4.23	4.22	4.16	4.19	4.22

3.2.5.9 Impact on Model Performance of Hyperparameter Tuning for Embedding Assignment Module

In this section, we examine the influence of hyperparameter selection within the embedding assignment module on the overall performance of the model, as shown in Table 3.13. Specifically, we focus on three key hyperparameters associated with the embedding assignment module, conducting our analysis using the validation set from the LibriMix dataset in max mode. We assess the impact of varying the hyperparameter values p_{sel}, p_{em}, and p_{res}, which are assigned values of 0.1, 0.3, 0.5, 0.7, and 0.9, while keeping the two other hyperparameters fixed to their default value, across multiple configurations where m is set to 1, 2, and all.

The experimental results demonstrate that the influence of the three hyperparameters on model performance varies depending on the value of m. For instance, for hyperparameter p_{sel}, larger values lead to better results when m equals 1, whereas smaller values yield better performance when m equals all. After carefully evaluating the overall performance across different values of m, we selected a set of parameters with $p_{sel} = 0.5$, $p_{em} = 0.3$, and $p_{res} = 0.9$ for relatively good performance.

3.2.6 Summary

This section proposed USED, a unified network for universal speaker extraction and diarization, which integrates speaker extraction and diarization for managing speech mixtures with varying overlap ratios and variable number of speakers. An embedding assignment module was designed to support producing results for a variable number of speakers based on speech references and enhance the model's robustness. Additionally, a multi-task interaction module was designed to leverage information from both speaker extraction and diarization tasks. Experimental results demonstrated significant improvements in both highly and sparsely overlapped speech scenarios for the speaker extraction and speaker diarization tasks.

□ End of chapter.

Chapter 4

Perception Beyond Words in Spoken Language Models

Summary

This chapter combines two threads of work on speech-centric LLMs and evaluation. The first section presents SD-Eval, a multidimensional benchmark for spoken dialogue understanding beyond words, with paralinguistic and environmental perspectives. The second section presents SOLLA, a speech-oriented LLM that integrates acoustic context via an audio tagging module and ASR-assisted prediction, together with the SA-Eval benchmark.

The previous chapters focused on how speech can be effectively represented and robustly processed under realistic acoustic conditions. These efforts address a necessary prerequisite for practical spoken systems: the ability to encode speech signals and remain robust in the presence of overlap and interference. However, robust signal processing alone is not sufficient for natural spoken interaction. Human speech

conveys much richer information than lexical content alone, including paralinguistic cues, speaker characteristics, and surrounding acoustic context.

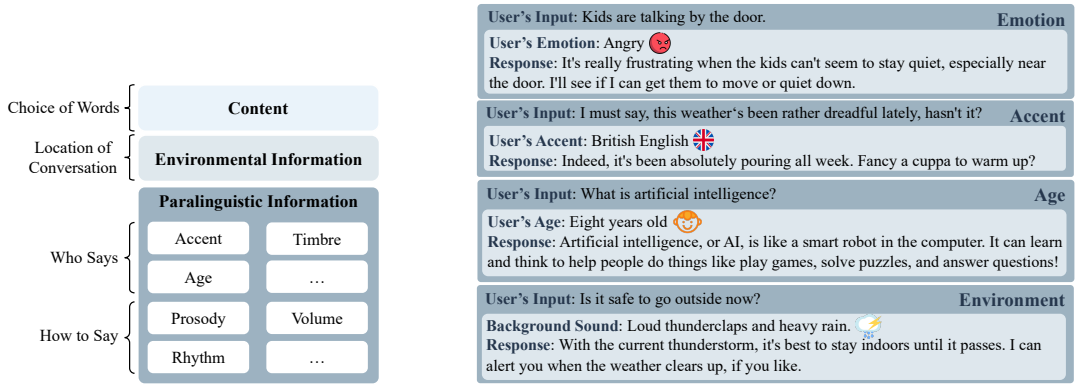
Motivated by this observation, this chapter shifts the focus from robust speech modeling to beyond-word perception in spoken language models. Rather than treating speech merely as an acoustic signal to be transcribed, this chapter investigates how spoken systems can perceive and utilize information beyond text. It first studies evaluation from the perspective of spoken dialogue understanding beyond words, and then examines how speech-oriented large language models can explicitly model acoustic context. This chapter therefore marks the dissertation's transition from foundational and robust speech modeling to higher-level cognitive spoken perception.

4.1 SD-Eval: A Benchmark for Spoken Dialogue Understanding Beyond Words

4.1.1 Introduction

Speech is a rich carrier of information and a central medium in human-computer interaction [Cohen and Oviatt, 1995; Cowie et al., 2001; Nass and Brave, 2005]. Its role in interaction extends beyond lexical content, as spoken signals also encode paralinguistic and environmental cues that can substantially shape the course and interpretation of conversation. In this dissertation, the information conveyed by speech is considered along three dimensions, as illustrated in Figure 4.1a: content information, environmental information, and paralinguistic information.

Among these dimensions, *content* information corresponds to the '**choice of words**", capturing the explicit meaning and linguistic structure of speech. *Environmental information* refers to the '**location of conversation**", including factors such as background noise and situational context that may affect how speech is in-



(a) Speech carries rich information including linguistic, para-linguistic and environmental information

(b) Examples of spoken dialogues impacted by the rich information carried in speech (e.g. emotion, accent, age, environment).

Figure 4.1: (a) Information embedded in speech: content, environmental, and paralinguistic information. (b) Examples of spoken dialogue, which illustrate the impact of user emotions, accents, age, and environmental information on the responses.

terpreted. *Paralinguistic information* encompasses non-verbal aspects that convey additional meaning and is further divided into '**who says**" and '**how to say**". The former includes properties such as accent, age, and timber of the speaker, all of which can influence speech perception and understanding, while the latter includes prosody, volume, and rhythm, which together characterize the expressive qualities of an utterance. Taken together, these dimensions underscore that spoken dialogue extends well beyond words alone and involves a broad range of interacting information sources. Figure 4.1b further shows how environmental and paralinguistic factors, including emotion, accent and age, affect responses.

Large Language Models (LLMs) have emerged as a universal interface for general-purpose assistance and have demonstrated remarkable capabilities across a wide range of settings [Achiam et al., 2023; Team et al., 2023; Touvron et al., 2023a,b; Zhang et al., 2023a; Yang et al., 2024a; Team et al., 2024; Young et al., 2024]. More recently, their scope has expanded from text-only processing to the understanding of multimodal inputs such as speech and image [Liu et al., 2024a; Zhu et al., 2023a; Zhang

et al., 2023b; Chu et al., 2023; Tang et al., 2024; Hu et al., 2024; OpenAI, 2024], thereby broadening the range of tasks they can support. In the speech domain, these models are primarily developed to perceive speech and to perform tasks specified through text instruction prompts. Such a formulation allows them not only to recognize content but also to access additional information in the signal, enabling speech-related tasks such as speech recognition and gender classification. However, in the absence of clear principles for task definition and model development, these systems often remain unable to generate appropriate responses directly from speech input. Advancing Speech LLMs therefore requires open-source datasets and evaluation metrics that reflect the full richness of information carried by speech.

To support this need, this dissertation introduces *a novel benchmark dataset for multidimensional evaluation of spoken dialogue understanding beyond words, namely SD-Eval*. The purpose of the dataset is to promote the development of more empathetic and intelligent spoken dialogue systems that can produce appropriate responses by incorporating paralinguistic and environmental information. The broader objective of SD-Eval is to serve as a benchmark for speech-to-speech conversation system development, while its initial version focuses on speech-to-text dialogue as a first step. This initial version contains four sub-tasks, each designed to evaluate responses to input utterances that vary in emotion, accent, age, and background sound. These sub-tasks are built from eight public datasets containing real-recorded speeches. More specifically, SD-Eval consists of four subsets, *test-emo*, *test-acc*, *test-age*, and *test-env*, corresponding respectively to emotion, accent, age and background sound, and contains 7,303 utterances with a total duration of 8.76 hours.

To assess the SD-Eval benchmark dataset, three different models are implemented, and a training set is constructed using a process similar to that used for SD-Eval itself. This training set contains 1,052.72 hours of speech data and 724.4k utterances. In addition, an empirical study is conducted on evaluation metrics for generated re-

sponses, covering objective evaluation methods such as BLEU and ROUGE, subjective opinion score, and LLM-based metrics.

4.1.2 Related Work

Spoken Conversation Datasets with Paralinguistic Label Paralinguistic information is crucial for comprehending speech and generating responses in spoken dialogues. Many speech emotion datasets are constructed under spoken dialogue scenarios, such as IEMOCAP [Busso et al., 2008], SEMAINE [McKeown et al., 2010], and MELD [Poria et al., 2019]. However, their primary purpose is to identify emotions in speech. Consequently, the dialogue data from these datasets is relatively less suited for training a spoken dialogue system.

Some recent studies build novel datasets such as E-chat200 [Xue et al., 2023] and StyleTalk [Lin et al., 2024], which are designed for spoken dialogue with a focus on emotional information. Nevertheless, the text and speech in these datasets are generated using ChatGPT and text-to-speech (TTS) models. Our dataset is based on a mixture of real-recording and synthesized speech and focuses on multiple aspects, including accents, emotions, ages, and background sounds.

Spoken Question Answering The spoken question answering (SQA) task requires the system to answer questions from speech. The past approaches [Tseng et al., 2016; Su and Fung, 2020] mainly divided this task into two parts through a cascaded model: automatic speech recognition (ASR) and text question answering. Recently, some systems [You et al., 2022; Nachmani et al., 2023] aim to achieve end-to-end spoken question answering.

Datasets in the field of SQA include Spoken SQuAD [Li et al., 2018], SCQA [You et al., 2022], HeySQuAD [Wu et al., 2023], OpenSAQA [Gong et al., 2023b], e.g. These datasets lack annotations of paralanguage information. StyleTalk [Lin et al., 2024]

provides annotations of speaking styles. Our work focuses more on paralinguistic and environmental information to simulate more realistic dialogue scenarios.

Evaluation Metrics for Open-Ended Generation Tasks Assessing the quality of text produced by language models or human authors for open-ended generation tasks has always been a difficult task. Traditional evaluation metrics such as BLEU [Papineni et al., 2002] and ROUGE [Lin, 2004] are based on the n-grams to measure the similarity between model outputs and references, while these metrics focus on lexical overlap, which is ineffective for open-ended generation problems. In addition, they show a relatively weak correlation with human judgement [Novikova et al., 2017]. Embedding-based metrics, such as BERTScore [Zhang* et al., 2020], use word or sentence embeddings to measure semantic similarity based on the references.

However, the answers to these tasks are open-ended without standard references, while collecting human preferences can be costly and laborious. Recently, several works [Liu et al., 2023c; Fu et al., 2023; Zheng et al., 2024] try to use LLMs for evaluating the responses of chat assistants, which shows a high correlation with human judgement. In our work, we adapt these LLM-based methods for spoken dialogue generation, with a focus on paralinguistic and environmental information.

4.1.3 SD-Eval Benchmark Dataset

4.1.3.1 Dataset Construction

SD-Eval is divided into four subsets: *test-emo*, *test-acc*, *test-age*, and *test-env*. Each subset focuses on a specific aspect: emotion, accent, age, and environment, respectively. The ultimate aim of SD-Eval is to create a benchmark dataset for the evaluation of speech-to-speech conversation systems. As a preliminary step, SD-Eval concentrates on speech-to-text dialogues. We construct SD-Eval through the following steps.

Data Collection As shown in Table 4.1, we select data from 8 public datasets to construct SD-Eval. For *test-emo* subset, RAVDESS [Lotfian and Busso, 2019], MEAD [Wang et al., 2020b], and JL Corpus [James et al., 2018] are selected as they contain audios with the same content but different emotions. For *test-env* subset, we choose real-recording speeches from the LibriSpeech [Panayotov et al., 2015b] test-clean subset and add background sounds using audio samples from AudioCaps [Kim et al., 2019].

Table 4.1: Statistics of the SD-Eval benchmark dataset, which includes four types of paralinguistic and environmental information.

Type	# Hours	# Utts	Constructed From	Labels
Emotion (*test-emo*)	1.11	1,289	RAVDESS [Lotfian and Busso, 2019], MEAD [Wang et al., 2020b], JL Corpus [James et al., 2018]	Sad, Angry, Fear, Disgust, Happy
Accent (*test-acc*)	5.34	4,310	VCTK [Yamagishi et al., 2019], Common Voice [Ardila et al., 2020]	England, Scottish, Northern Irish, Welsh, Irish, American, Canadian, Australian, New Zealand
Environment (*test-env*)	0.74	690	LibriSpeech [Panayotov et al., 2015b], AudioCaps [Kim et al., 2019], Synthesised Speech	Driving, Children's Voice, Sea Beach, Raining or Thundering, Bells, Sports Center, Bus or Subway
Age (*test-age*)	1.57	1,014	MyST [Pradhan et al., 2024], Synthesised Speech	Adult, Child
Summary	8.76	7,303	-	-

Synthetic Data Generation For *test-age* and *test-env*, a portion of the data is synthesized. For *test-age*, we use an internal zero-shot TTS model, which is trained on Libri-light, to generate speech data from the text in MyST [Pradhan et al., 2024] with adult speakers. For each text, we randomly select a sample from the LibriSpeech test-clean subset [Panayotov et al., 2015b] as the prompt to synthesize the data. For *test-env*, we first select audio collections corresponding to seven types of environments from AudioCaps [Kim et al., 2019]. Then, we mix each speech sample in the subset of LibriSpeech test-clean with audio randomly selected from these collections corresponding to each environmental scene. Simultaneously, we utilize GPT-4-Turbo [Achiam et al., 2023] to generate dialogue data for these seven scenarios and employ the TTS model to generate speech, forming part of the *test-env* subset.

Label Normalization Due to the varying number of label categories across different datasets, we first normalize the labels of all datasets. Specifically, labels of *test-acc* include *nine* widely used and representative accents: England, Scottish, Irish, Welsh, Northern Irish, American, Canadian, Australian, and New Zealand. For the *test-emo* subset, we firstly utilize Ekman's emotion model [Ekman and Friesen, 1971] as the labels, which contain neutral, surprise, sad, happy, angry, disgust, and fear, which are the basic emotions. We choose Ekman's emotion model because it is widely used in speech emotion recognition task [Busso et al., 2008; Lotfian and Busso, 2019; Wang et al., 2020b], ensuring that each category of emotion is well-represented and encompasses a substantial amount of data.

We then further exclude utterances with neutral and surprise emotions. Neutral implies that the speech does not convey positive or negative feelings, making the response primarily content-dependent. However, our focus is on examining the impact of speech emotion on text responses. Similarly, surprise can be associated with different sentiments, depending on the context [Sailunaz and Alhajj, 2019]. Therefore, we excluded data related to these two emotions. As a result, the *test-emo* subset includes five types of emotions: sad, happy, angry, disgust, and fear.

For the *test-env* subset, we select seven representative scenarios in daily life to serve as background sounds, as illustrated in Table 4.1. For the *test-age* subset, we focus on evaluating whether the model could generate comprehensible responses appropriate to different age groups. Consequently, the labels are divided into two categories: child and adult.

Data Filtering We filter the test data from three aspects. Firstly, some utterances of the four subsets are identified with notable ambiguity, potentially due to a lack of contextual information. To address this, we design a prompt and use GPT-4-turbo [Achiam et al., 2023] for automatic filtering, as illustrated in Figure 4.2. Following this

initial filtering, three human annotators are then required to evaluate the remaining utterances further using the same criteria as the prompt. Secondly, it is observed that some utterances within the *test-env* contain incorrect background sounds, possibly due to the multi-class labelling of the AudioCaps [Kim et al., 2019]. These utterances are subsequently identified and filtered by human annotators. Finally, we exclude utterances of *test-emo* subset where both the sentiment of transcript and emotion are positive or negative, aiming to enhance the impact of emotions on responses. For this purpose, a pre-trained sentiment classification model[1] is employed to predict the sentiments of utterances.

```
[System]
Evaluate the clarity and feasibility of a specified sentence for processing by
ChatGPT. Your response should consist of two parts:
1. State 'Yes' if the sentence is clear and feasible, or 'No' if it is
   ambiguous or vague.
2. Explain your assessment.
Ensure your response adheres strictly to JSON format with two keys: "answer"
and "reason", corresponding to your answers for the above questions.

[Sentence for Evaluation]
{input_sentence}
```

Figure 4.2: The prompt for filtering utterances.

Punctuation Restoration To improve response quality when using ChatGPT to generate utterance responses for datasets lacking punctuation, we apply a punctuation restoration model to the transcripts of MEAD, LibriSpeech, and the UK-Ireland datasets.

Response Generation Finally, we use GPT-4o [OpenAI, 2024] to generate five diverse responses for each utterance in SD-Eval by considering the content and emotion, accent, age or background sounds of speech signals. For instance, the prompt

[1] huggingface.co/lxyuan/distilbert-base-multilingual-cased-sentiments-student

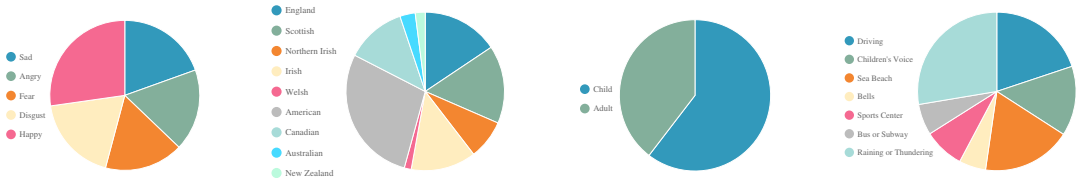
used to generate responses for utterances related to emotion is presented in Figure 4.3. All the prompts used to generate responses are included in Section 4.1.5.3.

```
[System]
Let's simulate a conversation between a hypothetical speaker and ChatGPT. I
need you to:
1. Create **five** diverse responses, each consisting of two or more sentences,
in reaction to the speaker's statement. Each response should appropriately
reflect the context and content provided by the speaker.
2. Assign an emotion selected from **joy, sadness, fear, anger, surprise,
neutral and disgust** to each response, with the language of the reply
demonstrating this emotion.

Format each of your responses with XML tags, such as <reply>reply</reply> and
<reply_emotion>reply emotion</reply_emotion>, which are corresponding to the
tasks above.

[Simulated Speaker's Statement]
{input_statement}

[Speaker's Emotion]
{emotion}
```

Figure 4.3: The prompt for generating responses of utterances related to emotion.

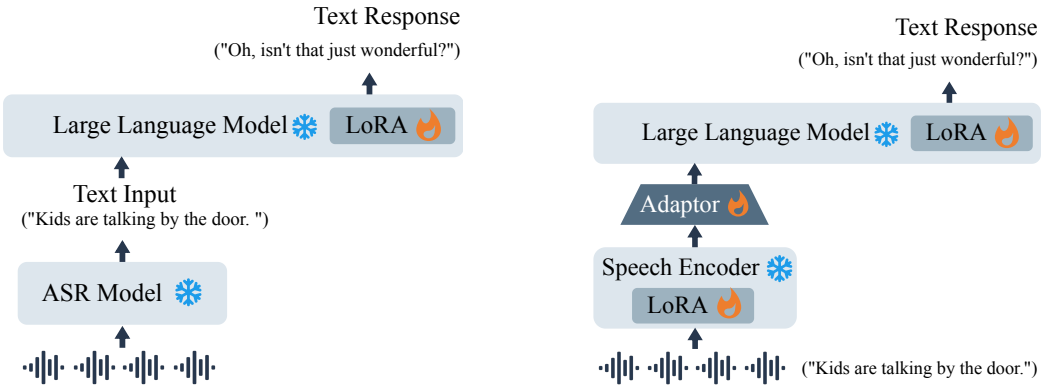
4.1.3.2 Dataset Statistics

The statistics of SD-Eval are presented in Table 4.1. The SD-Eval dataset comprises a total of 7,303 sentences and 8.76 hours of speech data. It contains three types of paralinguistic information (i.e. emotion, accent, age), and the environment type contains seven categories of environmental sounds. The pie charts in Figure 4.4 illustrate the data distribution for each category within each test set.



(a) test-emo (b) test-acc (c) test-age (d) test-env

Figure 4.4: Pie charts illustrating the data distribution for each category within each subset.



(a) Cascade LLM. (b) VS-LLM.

Figure 4.5: (a) Model Structure of Cascade LLM, which generates text response directly based on the ASR output. (b) Model structure of Vanilla Speech LLM (VS-LLM). The LLM takes speech representation as input, which is generated from a speech encoder and adaptor.

4.1.4 Benchmark Experiments

4.1.4.1 Training Set

To assess the SD-Eval benchmark dataset, we construct a training dataset from eleven open datasets for training models. We follow a procedure similar to SD-Eval, with the following two exceptions. Firstly, we simplify the data filtering process by removing sentences with inadequate and ambiguous labels. Secondly, we generated only one response for each sentence. The details, including data statistics and prompts, are introduced in Section 4.1.5.

4.1.4.2 Models

We implement several baselines trained using the proposed training set, aiming to evaluate their capability of comprehending the content of the speech, as well as recognizing emotions, accents, age, or background sounds. The implementations are detailed as follows.

Cascade LLM As shown in Figure 4.5a, the Cascade LLM consists of an automatic speech recognition (ASR) model to recognize the content, followed by an LLM to generate a response based on the text input. The ASR model is Whisper large-v3 [Radford et al., 2023], which is trained with a large amount of weakly supervised data for speech recognition and translation. The LLM is the InternLM2 chat model with 7 billion parameters (InternLM2-chat-7b) [Cai et al., 2024]. During training, the LLM is frozen, while we add a trainable LoRA adaptor [Hu et al., 2021] to facilitate model finetuning. We use this model as a baseline to evaluate responses if only knowing the content of the speech.

VS-LLM To understand and perceive content as well as paralinguistic and environmental information directly from speech, we design an end-to-end model named Vanilla Speech LLM (VS-LLM). As shown in Figure 4.5b, it consists of a speech encoder, an LLM and an additional adaptor to connect the speech encoder and LLM. The encoder of Whisper large-v3 [Radford et al., 2023] is used as the speech encoder, followed by a trainable adaptor to further down-sample the speech representation from the speech encoder. The adaptor comprises two linear layers, where the first linear layer is succeeded by a GELU activation function [Hendrycks and Gimpel, 2016], while the second one is followed by a two-dimensional average pooling operation for down-sampling. Similar to Cascade LLM, a frozen InternLM2-chat-7b with a trainable LoRA adaptor is employed as the LLM.

LLM (Upper Bound) To assess the system's upper bound upon the speech transcript, we also provide paralinguistic or environmental information as an additional label to the frozen InternLM2-chat-7b with LoRA for model finetuning. The input format is a concatenation of ground-truth transcripts and labels. For instance, *"How are you?<Emotion:Happy>"* is the input of an utterance. The transcript of this utterance is "How are you?" The emotion contained in this utterance is happy.

Besides the above self-implemented models, we further assess the performance of the off-the-shelf speech LLM models on SD-Eval. All text instruction prompts for open-sourced models can be found in Section 4.1.5.6.

Qwen-Audio Qwen-Audio [Chu et al., 2023] is designed for handling a wide range of audio types and tasks. The model scales up pre-training across more than 30 tasks, including speech, natural sounds, and music, in multiple languages, achieving strong performance without task-specific fine-tuning. Since Qwen-Audio requires a text instruction prompt for each input to define the task, we add a text instruction prompt to let the model generate a text response based on the speech.

Qwen2-Audio Qwen2-Audio [Chu et al., 2024] is a work based on Qwen-Audio. It is trained on larger-scale data and employs DPO [Rafailov et al., 2024] for optimising models to follow human preferences. Unlike Qwen-Audio, Qwen2-Audio has two modes: voice chat (Qwen2-Audio-VC) and audio analysis (Qwen2-Audio-AA). In Audio Analysis mode, it can analyze various audio types and identify command segments within the audio. In Voice Chat mode, it acts like a conversational agent, allowing users to engage in dialogue through audio. We also utilize a text instruction prompt for audio analysis mode during evaluation.

SALMONN SALMONN [Tang et al., 2024] is a multi-modal large language model designed to process and understand general auditory inputs, including speech, audio events, and music. SALMONN integrates a pre-trained text-based large language model (LLM) with dual auditory encoders — Whisper [Radford et al., 2023] for speech and BEATs [Chen et al., 2022c] for non-speech audio. We use a predefined text prompt for evaluation.

4.1.4.3 Evaluation Metrics

Objective Evaluation We propose a reference-free metric using the LLMs for response evaluation. Specifically, we design different prompts for the evaluations of each subset. By the prompts, the LLM judge must consider (a) the response's naturalness, coherence, engagingness and groundedness. (b) Whether the response is appropriate and fully considers the emotion, accent, age or background sound of input speech. The LLM judge is then asked to directly assign a score, such as 5 on a 1 - 10 scale, to a single answer. For comparison, we further include the results of n-gram-based metrics, such as ROUGE-L [Lin, 2004], BLEU-4 [Papineni et al., 2002] and METEOR [Banerjee and Lavie, 2005], and embedding-based metrics, such as BERTScore [Zhang* et al., 2020] [2].

Subjective Evaluation In addition, we conduct a human evaluation on 200 randomly selected utterances from the four subsets, with each subset contributing 50 utterances. Each sample is assessed by at least three human evaluators, who are instructed to rate the generated responses. Each utterance has three samples, corresponding to three utterance-response pairs generated by Cascade LLM, VS-LLM, and LLM (Upper Bound), respectively. We ensure that each valid sample is evaluated by at least three human annotators. Consequently, each subset has no fewer than 120 valid samples.

4.1.4.4 Experimental Setup

Configuration for Model Training All self-implemented models are built using xtuner [Contributors, 2023] and optimized with AdamW [Loshchilov and Hutter, 2017] at a learning rate of 2×10^{-4}. The models are finetuned on 16 A100 GPUs, each with a batch size of 16, for two epochs. For the LoRA adaptor of the InternLM2-

[2] We use Hugging Face Evaluate for scoring and the BERT model is *roberta-large*.

chat-7b model, we use a rank of 512 and α of 256. In contrast, for the encoder of the Whisper large-v3 model, the rank is set to 64 and α to 16.

Inference Setting of LLM Judge In addition to GPT-4o, we employ Yi-1.5-34B-Chat [3] [Young et al., 2024], Qwen2-57B-A14B-Instruct [4] [Yang et al., 2024a], and gemma-2-27B-it [5] [Team et al., 2024], as LLM judges. To speed up inference and save memory, we utilize llama.cpp [6] for LLM inference. Specifically, we use a quantized version Q6_K of these two models to achieve a balance between efficiency and performance. This configuration allows using CPUs or only one A100 GPU for evaluation.

4.1.4.5 Main Results

Table 4.2 shows the main results of all models on SD-Eval. Firstly, across all four test sets, VS-LLM outperformed Cascade LLM on all metrics. This indicates that using speech as a direct input allows VS-LLM to implicitly learn paralinguistic and environmental information. Secondly, the performance of VS-LLM is inferior to that of LLM (Upper Bound). The main reason may be that VS-LLM implicitly acquires content as well as paralinguistic and environmental information directly from speech, whereas the LLM (Upper Bound) utilizes ground truth transcripts and labels. This indicates that the way to process the input data is important for model performance. A detailed ablation study regarding the input data will be introduced later.

As for the open-sourced models, while SALMONN [Tang et al., 2024] achieves better or comparable performance compared to Qwen2-Audio-AA [Chu et al., 2024] and Qwen-Audio [Chu et al., 2023], Qwen2-Audio-VC [Chu et al., 2024] performs much better than other open-sourced models as voice chat mode is more suitable for

[3] https://huggingface.co/bartowski/Yi-1.5-34B-Chat-GGUF
[4] https://huggingface.co/Qwen/Qwen2-57B-A14B-Instruct-GGUF
[5] https://huggingface.co/bartowski/gemma-2-27b-it-GGUF
[6] https://github.com/ggerganov/llama.cpp

Table 4.2: Main results of five models on four subsets of SD-Eval. † The scores from human evaluations are calculated based on randomly sampled data as described in Section 4.1.4.3.

Model	BLEU-4	ROUGE-L	METEOR	BERTScore	LLM Judges				Human
					Yi-1.5	Qwen2	Gemma	GPT-4o	Evaluation †
test-emo / Emotion									
SALMONN [Tang et al., 2024]	2.48	16.57	18.97	86.20	4.98	3.35	2.32	2.61	-
Qwen-Audio [Chu et al., 2023]	3.93	19.02	16.82	86.59	4.19	2.35	2.02	2.24	-
Qwen2-Audio-AA [Chu et al., 2024]	3.01	16.82	17.51	86.17	4.75	2.52	2.21	2.33	-
Qwen2-Audio-VC [Chu et al., 2024]	2.21	14.57	22.08	85.41	5.88	3.83	2.93	3.25	-
Cascade LLM	4.66	21.98	21.70	87.93	5.67	3.86	2.35	4.47	5.05
VS-LLM	8.29	25.52	27.23	89.48	6.40	4.56	4.03	5.30	6.31
LLM (Upper Bound)	**12.35**	**26.08**	**28.27**	**89.77**	**7.03**	**5.82**	**6.46**	**6.74**	**7.29**
test-acc / Accent									
SALMONN[Tang et al., 2024]	7.50	22.22	21.23	87.53	5.27	6.16	3.16	2.93	-
Qwen-Audio [Chu et al., 2023]	4.52	17.15	17.78	85.59	3.48	3.45	1.86	1.72	-
Qwen2-Audio-AA [Chu et al., 2024]	7.26	21.80	19.68	87.68	5.04	6.13	3.01	2.54	-
Qwen2-Audio-VC [Chu et al., 2024]	3.47	17.46	23.77	86.26	5.96	6.20	3.94	4.37	-
Cascade LLM	14.51	30.53	34.13	89.66	7.23	7.32	5.65	6.62	6.71
VS-LLM	17.98	33.06	37.65	90.08	7.82	7.65	6.59	7.85	7.95
LLM (Upper Bound)	**18.35**	**33.48**	**38.27**	**90.23**	**7.85**	**7.75**	**6.73**	**8.02**	**8.30**
test-age / Age									
SALMONN[Tang et al., 2024]	10.03	24.95	23.55	88.10	5.41	4.66	3.14	3.35	-
Qwen-Audio [Chu et al., 2023]	7.28	23.09	21.80	86.72	4.43	3.98	2.25	2.50	-
Qwen2-Audio-AA [Chu et al., 2024]	6.81	22.72	20.51	87.47	5.19	4.58	3.01	3.14	-
Qwen2-Audio-VC [Chu et al., 2024]	5.64	18.90	28.23	86.70	7.03	5.92	4.48	5.06	-
Cascade LLM	15.36	31.96	31.99	90.08	7.22	7.16	6.46	4.47	6.51
VS-LLM	17.22	34.17	33.78	90.63	7.74	7.39	7.25	7.95	7.11
LLM (Upper Bound)	**18.78**	**35.62**	**36.01**	**91.00**	**7.82**	**7.54**	**7.40**	**8.25**	**7.44**
test-env / Environment									
SALMONN[Tang et al., 2024]	2.87	16.53	21.37	86.71	4.70	5.00	3.40	3.56	-
Qwen-Audio [Chu et al., 2023]	2.37	16.83	17.50	85.81	3.77	1.86	2.16	2.14	-
Qwen2-Audio-AA [Chu et al., 2024]	2.97	16.32	19.84	86.50	4.52	5.02	3.49	3.50	-
Qwen2-Audio-VC [Chu et al., 2024]	2.06	12.35	23.40	85.17	6.30	6.21	4.85	5.30	-
Cascade LLM	5.44	21.75	26.41	88.22	6.03	5.84	5.31	5.66	6.62
VS-LLM	9.42	25.85	28.27	89.23	6.14	5.88	5.10	5.82	7.11
LLM (Upper Bound)	**11.72**	**27.95**	**31.50**	**89.73**	**7.14**	**7.14**	**6.25**	**7.40**	**8.13**

conversation. However, the performance of open-sourced models in SD-Eval is not very impressive. This suggests a current lack of well-defined tasks and datasets in this area. To improve the model's performance, future directions may include using larger-scale and more diverse data [He et al., 2024; Kahn et al., 2020; Zhang et al., 2022], as well as employing methods such as disentanglement [Ju et al., 2024; Zhang et al., 2024] to enhance the model's understanding of speech information.

4.1.4.6 Analysis

Ablation Study of Input Data We further conduct an ablation study in terms of the input data, as shown in Table 4.3. We investigate several models with different inputs. Among them, Model 1, which belongs to Speech LLM and is without any text input, refers to VS-LLM. Model 4 utilizing transcripts from the ASR model as input is Cascade LLM. Additionally, Model 8 uses ground truth transcripts and labels, which is LLM (Upper Bound). For ASR and speech emotion recognition (SER), the models are Whisper large-v3 [Gong et al., 2023a] and emotion2vec [Ma et al., 2023b] [7].

Firstly, we examine the effect of content quality. We observe that the performance of models utilizing ASR-generated transcripts (Model 5 and Model 7) is inferior across all metrics compared to their counterparts (Model 6 and Model 8) that use ground-truth transcripts. Next, we examine the effect of emotion label quality. For the LLM-based system, models using emotion labels from the SER model (Model 5 and Model 6) perform worse across all metrics compared to those using ground-truth labels (Model 7 and Model 8). Model 4, which is trained without emotion labels, performs the worst. A similar trend is observed in the models of Speech LLM, where Model 2 obtained emotion labels from the SER model outperforms Model 1, while Model 3, trained with ground truth labels, achieves the best performance among all three models. This corroborates our hypothesis in Section 4.1.4.5.

Correlations between Objective Metrics and Human Evaluation Finally, we investigate the correlations between scores of objective metrics and human evaluation, as shown in Table 4.4. Following the configuration of GPTScore [Fu et al., 2023], we utilize dataset-level Spearman and Kendall-Tau correlation metrics. Firstly, the experimental results indicate that all LLM judges exhibit a significantly higher correlation with human evaluations compared to other metrics. Secondly, as an LLM

[7] https://huggingface.co/emotion2vec/emotion2vec_plus_seed

Table 4.3: Ablation study on *test-emo* subset. The model types include LLM (text input only) and Speech LLM (text and speech inputs). "Trans" refers to the method used to obtain the transcripts. Options include "ASR" (generated by an ASR model) and "GT" (ground-truth transcript). "Emotion Label" indicates the source of the speech emotion label for the utterance, either "SER" (produced by a speech emotion recognition model) or "GT" (ground-truth label). "N/A" means the input is not used for the model.

Index	Model Type	Trans	Emotion Label	BLEU-4	ROUGE-L	METEOR	BERTScore	LLM Judge			
								Yi-1.5	Qwen2	Gemma	GPT-4o
1		N/A	N/A	8.29	25.52	27.23	89.48	6.40	4.56	4.03	5.30
2	Speech LLM	N/A	SER	10.37	27.29	28.59	89.81	6.76	5.26	4.37	6.11
3		N/A	GT	10.21	27.22	28.45	89.85	6.96	5.45	4.45	6.41
4		ASR	N/A	4.66	21.98	21.70	87.93	5.67	3.86	2.35	4.47
5		ASR	SER	11.37	26.03	27.66	89.66	6.74	5.35	5.62	6.13
6	LLM	GT	SER	11.85	26.05	27.78	89.75	6.85	5.53	6.07	6.38
7		ASR	GT	11.85	26.03	28.19	89.68	6.96	5.64	6.04	6.47
8		GT	GT	**12.35**	**26.08**	**28.27**	**89.77**	**7.03**	**5.82**	**6.46**	**6.74**

judge, GPT-4o consistently achieves the best or second-best performance in most cases. These findings strongly validate the effectiveness of LLM judges as evaluation metrics.

Table 4.4: Spearman (ρ) and Kendall-Tau (τ) correlations between human evaluation and different metrics.

Metrics	test-emo		test-acc		test-age		test-env		Overall	
	ρ	τ	ρ	τ	ρ	τ	ρ	τ	ρ	τ
BLEU-4	0.211	0.170	0.197	0.156	0.316	0.232	0.169	0.136	0.208	0.160
ROUGE-L	0.203	0.140	0.263	0.192	0.318	0.215	0.216	0.144	0.258	0.176
METEOR	0.249	0.168	0.272	0.194	0.350	0.242	0.315	0.214	0.336	0.229
BERTScore	0.378	0.269	0.252	0.184	0.321	0.216	0.286	0.199	0.291	0.199
Yi-1.5	0.641	0.493	0.435	0.345	0.356	0.289	0.558	0.441	0.492	0.381
Qwen2	0.618	0.456	0.198	0.161	0.347	0.278	0.449	0.352	0.474	0.362
Gemma	0.639	0.493	0.375	0.304	0.448	0.356	0.563	0.439	0.492	0.380
GPT-4o	**0.731**	**0.568**	**0.659**	**0.541**	**0.474**	**0.356**	**0.577**	**0.459**	**0.613**	**0.468**

4.1.5 Supplementary Details

4.1.5.1 Statistics of Training Set

Table 4.5 shows the statistics of training set. For training data related to the environment, we generate one response for each sentence, except for data related to the environment, which has five different responses for each sentence to serve the purpose of data augmentation.

Table 4.5: Statistics of training set. ChatGPT Version refers to the specific version of ChatGPT used to generate the data.

Type	# Hours	# Utts	Constructed From	Labels	ChatGPT Version
Emotion	120.60	100.5k	MSP-Podcast [Lotfian and Busso, 2019], IEMOCAP [Busso et al., 2008], MELD [Poria et al., 2019], EmoV-DB [Adigwe et al., 2018], ESD [Zhou et al., 2022], CREMA-D [Cao et al., 2014]	Angry, Contempt, Disgust, Fear, Happy, Neutral, Sad, Surprise,Frustrated Excited, Amused, Sleepiness	GPT-3.5-Turbo
Accent	759.75	508.6k	UK-Ireland dataset [Demirsahin et al., 2020], VCTK [Yamagishi et al., 2019], Common Voice [Ardila et al., 2020]	England, Scottish, Northern Irish, Welsh, Irish, American, Canadian, Australian, Nea Zealand	GPT-4o
Environment	32.06	47.1k	LibriSpeech [Panayotov et al., 2015b], AudioCaps [Kim et al., 2019], Synthesised Speech	Driving, Children's Voice, Sea Beach, Raining or Thundering, Bells, Sports Center, Shopping Center, Bus or Subway	GPT-4-Turbo
Age	140.31	73.2k	MyST [Pradhan et al., 2024]	Child	GPT-3.5-Turbo
Summary	1,052.72	729.4k	-	-	-

4.1.5.2 Zero-shot TTS Model

Our internal zero-shot TTS model is an auto-regressive model, which is similar to BASE-TTS [Łajszczak et al., 2024]. We evaluate our TTS model with some objective metrics. We assess objective metrics including speaker similarity (SIM-O and SIM-R), and robustness (WER) in the following ways: 1) To evaluate speaker similarity, we use the WavLM-TDCNN [Chen et al., 2022b] speaker embedding model. This model measures how closely generated samples match the original prompt (SIM-O) and the reconstructed prompt (SIM-R). 2) For measuring robustness, we calculate the Word Error Rate (WER) using a CTC-based HuBERT model[8] that was initially trained on Librilight and subsequently finetuned on the 960-hour training dataset from LibriSpeech. We compare our models with SOTA auto-regressive TTS models: VALL-E

[8] https://huggingface.co/facebook/hubert-large-ls960-ft

[Wang et al., 2023a], and CLaM-TTS [Kim et al., 2024], VoiceCraft [Peng et al., 2024], XTTS-v2[9], and WhisperSpeech[10]. We adapt classifier-free guidance (cfg) [Ho and Salimans, 2022; Sanchez et al., 2023] for better generation. We use LibriSpeech test-clean for evaluation, which contains 40 distinct speakers. Following [Wang et al., 2023a; Ju et al., 2024], we randomly select one sentence for each speaker as the target and a 3-second clip as the prompt from the same speaker's speech.

Table 4.6: Evaluation results of the zero-shot TTS model on LibriSpeech test-clean.

	Training Data	Sim-O↑	Sim-R↑	WER↓
Ground Truth	-	0.68	-	0.34
VALL-E	LibriLight	-	0.58	5.9
CLaM-TTS	MLS	0.49	0.54	5.11
VoiceCraft	GigaSpeech	0.45	-	6.68
XTTS-v2	-	0.51	-	5.5
WhisperSpeech	LibriLight	0.48	-	4.78
Ours	LibriLight	0.58	0.61	5.56
Ours (w. cfg)	LibriLight	0.60	0.63	4.32
Ours (w. cfg, rerank 5)	LibriLight	0.63	0.66	2.01

[9] https://huggingface.co/coqui/XTTS-v2
[10] https://github.com/collabora/WhisperSpeech

4.1.5.3 Prompts for Generating Responses

```
[System]
Let's simulate a conversation between a simulated speaker and you, ChatGPT. I
need your help to finish the following three tasks:
1. Generate your reply which consists of two or more sentences to the
simulated speaker. Your reply should be able to reflect the provided
information.
2. Select an appropriate emotion (happy, sad, fear, angry, surprise, neutral,
disgust) for your reply, which should be conveyed through the language used.
3. Explain how your reply reflects the provided information of the simulated
speaker. Your reason should be less than 3 sentences.

Your output must strictly adhere to JSON format with three keys: "reply",
"reply_emotion" and "reason" corresponding to your answers for the three tasks.

[What the simulated speaker said]
{input_statement}

[Emotion of the simulated speaker]
{emotion}
```

Figure 4.6: The prompt used to generate responses of utterances for training set related to emotion.

```
[System]
Let's simulate a conversation between a simulated speaker and you, ChatGPT. I
need your help to finish the following two tasks:
1. Generate a reply of two or more sentences to the simulated speaker. Ensure
that your reply mimics the provided information, including adopting a similar
accent style as the simulated speaker.
2. Explain how your reply reflects the provided information of the simulated
speaker. Your reason should be less than 3 sentences.

Your response must be formatted in JSON, with two keys: "reply" for the first
task and "reason" for the second task.

[What the simulated speaker said]
{input_statement}

[Accent of the simulated speaker]
{accent}
```

Figure 4.7: The prompt used to generate responses of utterances for training related to accent.

```
[System]
Simulate a conversation between a child and ChatGPT. Complete the following
tasks:
1. Generate a reply consisting of at least two sentences, tailored to the
child's age provided in the input.
2. Briefly explain (in less than three sentences) how your reply considers the
child's age.

Your output must strictly adhere to JSON format with three keys: "reply",
"reply_emotion" and "reason" corresponding to your answers for the three tasks.

[What the simulated child said]
{input_statement}

[Child's Age]
{age}
```

Figure 4.8: The prompt used to generate responses of utterances for training related to age.

```
[System]
You are tasked with simulating a conversation between a simulated speaker and
yourself, ChatGPT. Could you give responses based on a text with a certain
environmental sound? For the scenario, provide a brief description of the
background sound and **five** suitable model response that aligns with the
context.

Your response must be formatted in JSON, each entry should contain the
following keys: "reply", "reason".

[What the simulated speaker said]
{input_statement}

[Background Sound]
{background_sound}
```

Figure 4.9: The prompt used to generate responses of utterances for training related to background sound.

Prompts for Training Set.

```
[System]
Let's simulate a conversation between a hypothetical speaker and ChatGPT. I
need you to:
1. Create **five** diverse responses, each consisting of two or more sentences,
in reaction to the speaker's statement. Each response should appropriately
reflect the context and content provided by the speaker.
2. Assign an emotion selected from **joy, sadness, fear, anger, surprise,
neutral and disgust** to each response, with the language of the reply
demonstrating this emotion.

Format each of your responses with XML tags, such as <reply>reply</reply> and
<reply_emotion>reply emotion</reply_emotion>, which are corresponding to the
tasks above.

[Simulated Speaker's Statement]
{input_statement}

[Speaker's Emotion]
{emotion}
```

Figure 4.10: The prompt used to generate responses of utterances for *test-emo*.

```
[System]
Simulate a conversation between a hypothetical speaker and ChatGPT. Produce
**five** varied responses, each comprising at least two sentences. Ensure that
your reply mimics the provided information, including adopting a similar
accent style as the simulated speaker. Your output should be **strictly**
formatted for each response using XML tags, <reply> and </reply>.

[Simulated Speaker's Statement]
{input_statement}

[Speaker's Accent]
{accent}
```

Figure 4.11: The prompt used to generate responses of utterances for *test-acc*.

```
[System]
Simulate a conversation between a child and ChatGPT. Produce **five** varied
responses, each comprising at least two sentences, suited to the child's age
specified in the input. Your output should be **strictly** formatted for each
response using XML tags, <reply> and </reply>.

[Simulated Child's Statement]
{input_statement}

[Child's Age]
{age}
```

Figure 4.12: The prompt used to generate responses of utterances for *test-age*.

```
[System]
You are tasked with simulating a conversation between a simulated speaker and
yourself, ChatGPT. Could you give responses based on a text with a certain
environmental sound? For the scenario, provide a brief description of the
background sound and **five** suitable model response that aligns with the
context.

Your response must be formatted in JSON, each entry should contain the
following keys: "reply", "reason”.

[What the simulated speaker said]
{input_statement}

[Background Sound]
{background_sound}
```

Figure 4.13: The prompt used to generate responses of utterances for *test-env*.

Prompts for SD-Eval.

4.1.5.4 Prompts for LLM Evaluation

```
[System]
Please act as an impartial judge and evaluate the quality of the response
provided by an AI assistant to the user's statement and emotion displayed
below. Your evaluation should consider whether it contains an appropriate
sentiment with respect to the user's emotion. Please also consider factors
such as the naturalness, coherence, engagingness and groundedness of the
response. Please make sure you read and understand these instructions
carefully. Please be as objective as possible. Begin your evaluation by
providing a short explanation. After providing your explanation, please rate
the response on a scale of 1 to 10 by **strictly** following this format:
"Rating: [[rating]]", for example: "Rating: [[5]]".

[User's Statement]
{statement}

[User's Emotion]
{info}

[AI Assistant's Response]
{response}
```

Figure 4.14: The prompt for evaluating *test-emo* using LLM.

```
[System]
Please act as an impartial judge and evaluate the quality of the response
provided by an AI assistant to the user's statement and accent displayed below.
Your evaluation should consider whether the AI assistant recognizes the user's
accent correctly so that the response contains appropriate slang with respect
to the user's accent. Please also consider factors such as the naturalness,
coherence, engagingness and groundedness of the response. Please make sure you
read and understand these instructions carefully. Please be as objective as
possible. Begin your evaluation by providing a short explanation. After
providing your explanation, please rate the response on a scale of 1 to 10 by
**strictly** following this format: "Rating: [[rating]]", for example: "Rating:
[[5]]".

[User's Statement]
{statement}

[User's Accent]
{info}

[AI Assistant's Response]
{response}
```

Figure 4.15: The prompt for evaluating *test-acc* using LLM.

```
[System]
Please act as an impartial judge and evaluate the quality of the response
provided by an AI assistant to the user's statement and age displayed below.
Your evaluation should consider whether it contains an appropriate tone of
voice with respect to the user's age. Please also consider factors such as the
naturalness, coherence, engagingness and groundedness of the response. Please
make sure you read and understand these instructions carefully. Please be as
objective as possible. Begin your evaluation by providing a short explanation.
After providing your explanation, please rate the response on a scale of 1 to
10 by **strictly** following this format: "Rating: [[rating]]", for example:
"Rating: [[5]]".

[User's Statement]
{statement}

[User's Age]
{info}

[AI Assistant's Response]
{response}
```

Figure 4.16: The prompt for evaluating *test-age* using LLM.

```
[System]
Please act as an impartial judge and evaluate the quality of the response
provided by an AI assistant to the user's statement and background sound
displayed below. Your evaluation should consider whether it considers the
user's background sound and generates an appropriate response. Please also
consider factors such as the naturalness, coherence, engagingness and
groundedness of the response. Please make sure you read and understand these
instructions carefully. Please be as objective as possible. Begin your
evaluation by providing a short explanation. After providing your explanation,
please rate the response on a scale of 1 to 10 by **strictly** following this
format: "Rating: [[rating]]", for example: "Rating: [[5]]".

[User's Statement]
{statement}

[User's Background Sound]
{info}

[AI Assistant's Response]
{response}
```

Figure 4.17: The prompt for evaluating *test-env* using LLM.

4.1.5.5 Prompts for Generating Dialogue Data of *test-env*

```
[System]
You are tasked with simulating a conversation between a simulated
speaker and yourself, ChatGPT. Could you provide examples of how a
speech dialogue model would adapt its responses based on different
environmental sounds? Please include scenarios where the model detects
distinctive background noises (such as in a busy marketplace, at a beach,
children's voice, or during a thunderstorm). For each scenario, provide
a brief description of the background sound and a suitable model
response that aligns with the context. Let's first focus on a special
background noise type: {background_sound}. Your response must be
formatted in JSON, each entry should contain the four keys: "text",
"background", "reply", and "reason", "reason" should focus on why the
reply may change due to the presence of background sound. I will give
you some examples:

{text: "Can you tell me about the effects of adult beverages on health?",
background: "children's voice", reply: ["It sounds like there might be
little ears around! If you're asking for general health information,
it's important to maintain a balanced lifestyle, which includes
moderation in all things."], reason: "The large model can answer the
question about the effects of adult beverages, but because there are
children's voices in the background, it is not suitable for direct
answers, but for discussing more general health topics."}

{text: "What should I wear today?", background: "raining or thundering",
reply: ["Given the rainy and windy weather outside, I recommend wearing
a waterproof jacket and sturdy, water-resistant shoes. Would you like a
reminder to bring an umbrella?"], reason: "Since there are sounds of
rain and thunder in the background, it is recommended to bring an
umbrella."}

{text: "Where's a good place to eat around here?", background: "driving
or traffic", reply: ["Since you're on the road, I'll find something easy
to access. There's a highly-rated grill just off the next exit. Want me
to guide you there?"], reason: "..."}

OK, now it's your turn! Please provide examples of how a speech dialogue
model would adapt its responses based on different environmental sounds,
let's firstly focus on a special background types: {background_sound}.
In my examples, each "reply" contains only one response. You should
provide five proper responses for each example. Please generate five
examples. Each example contains five responses, and for each example,
only give one reason for general.
```

Figure 4.18: The prompt for generating dialogue data of *test-env*.

4.1.5.6 Text Instruction Prompts for Open-Sourced Models

- **Qwen-Audio:** "How to respond to the audio?"
- **Qwen2-Audio-AA:** "Suppose you are in a conversation, and this audio is what someone else is saying to you. Please respond to him/her directly without outputting the transcript."
- **SALMONN:** "Please answer the question in detail."

4.1.6 Summary

This chapter introduced SD-Eval, a benchmark dataset designed for the multidimensional evaluation of spoken dialogue understanding and generation. SD-Eval includes 7,303 utterances amounting to 8.76 hours of speech data, aggregated from eight public datasets, and focuses on paralinguistic and environmental information across four perspectives: emotion, accent, age, and background sound. The dataset aims to advance the creation of more empathetic and intelligent spoken dialogue systems capable of generating appropriate responses by considering paralinguistic and environmental information. Our comprehensive evaluation demonstrates that models conditioned with paralinguistic or environmental information outperform their counterparts in both objective evaluation and subjective evaluation. Furthermore, our experiments indicate that LLM-based metrics have a higher correlation with human evaluation compared to traditional metrics.

The limitations and future work for SD-Eval are as follows: First, SD-Eval accommodates only speech-to-text dialogues, limiting the evaluation of system responses at the text level. Second, SD-Eval currently supports the evaluation of single-turn dialogues only, limiting its application to more complex, multi-turn interactions. Finally, SD-Eval includes four sub-tasks that focus on speech elements such as emotion, accent, age, and environmental information. However, it does not yet account for other

aspects, such as the gender of the speaker. Addressing these aspects constitutes our future work, with the ultimate goal of developing a benchmark dataset capable of multidimensional evaluation for multi-turn speech-to-speech dialogues.

4.2 SOLLA: Towards a Speech-Oriented LLM That Hears Acoustic Context

4.2.1 Introduction

In everyday settings, people are surrounded by diverse sounds, including speech, ambient noise, and music, each of which conveys information that supports interaction with the environment. A simple exchange such as asking a friend in a forest, "Do you recognize this bird sound?" illustrates how naturally such auditory interpretation is embedded in daily life. More broadly, the capacity to interpret and respond to varied sound cues underpins both effective communication and situational awareness. A speech-oriented LLM is expected to play a similar role. From this perspective, enabling speech-oriented LLMs to integrate and understand heterogeneous auditory inputs is both a technical challenge and an important step toward more capable human-computer interaction.

LLMs have shown remarkable versatility as a universal interface for a wide range of assistance [Achiam et al., 2023; Team et al., 2023; Touvron et al., 2023a,b; Yang et al., 2024a; Team et al., 2024]. More recently, this generality has extended beyond text to multimodal inputs, including speech [Chu et al., 2023, 2024; Tang et al., 2024; Hu et al., 2024; OpenAI, 2024; Défossez et al., 2024; Fang et al., 2024; Xie and Wu, 2024], thereby enlarging the set of tasks that such models can address. Existing speech LLMs have reported strong performance on tasks such as speech recognition and speech translation, but they mainly emphasize understanding input signals and carrying out tasks

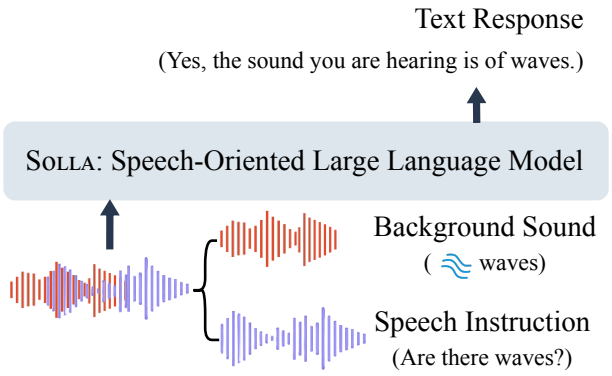


Figure 4.19: An illustration of SOLLA that a speech-oriented LLM can understand the acoustic context and generate the answer.

specified through text instructions [Chu et al., 2023, 2024; Tang et al., 2024; Hu et al., 2024]. At the same time, recent work has begun to investigate seamless voice interaction as a route toward more natural and lower-latency human-computer communication [Défossez et al., 2024; Fang et al., 2024; Xie and Wu, 2024]. Even so, comparatively little attention has been given to models that simultaneously understand audio and speech. Related efforts on audio analysis and processing [Kong et al., 2024; Gong et al., 2024; Deshmukh et al., 2023] likewise rely on audio encoders to derive audio features for LLM input, yet these systems still depend on text instructions to specify the task. *In a hand-free speech interaction, it is infeasible to provide text instructions.*

Understanding spoken instructions in the presence of acoustic background context, such as audio events, background speech, or music, remains difficult. The problem requires the model, first, to identify the question accurately from the speech instruction and, second, to formulate an appropriate response from the information contained in the surrounding acoustic scene. This difficulty is intensified in realistic conditions, where speech and background sound are often mixed in complex ways. Overlapping sources and different signal-to-noise ratios (SNR), for example, may prevent the model from extracting the key information carried by both the spoken in-

struction and the accompanying audio. Under such conditions, reliable answering may depend on disentangling these sources before response generation.

As an initial step toward joint understanding of spoken instructions and background acoustic context, this work concentrates on audio event context. Specifically, we **propose a SOLLA model, a speech-oriented LLM that hears acoustic context, and design a SA-Eval benchmark dataset**. To improve the interpretation of audio information from the input signal, we introduce an audio tagging module that predicts audio events and produces corresponding event representations. In addition, we present an ASR-assisted prediction task that enables the model to capture the spoken content before generating its response.

To assess the performance of SOLLA as well as publicly available models, we introduce SA-Eval, a benchmark built from publicly available datasets that comprises three tasks: audio event classification, audio captioning, and audio question answering. To reflect varying levels of difficulty in the speech instructions, SA-Eval is organized into two settings, *easy* and *hard*, for more comprehensive evaluation. In the *easy* mode test set, speech instructions and audio do not overlap. In the *hard* mode test set, by contrast, a more realistic condition is simulated in which speech instructions and audio overlap under low SNR, so as to evaluate whether the model can accurately capture speech instruction information in noisy environments.

4.2.2 Related Work

Speech/Audio LLMs Recent advancements in speech and audio LLMs have been significant. Numerous studies have investigated the development of speech/audio LLMs, which can be categorized into two types.

The first category of models [Chu et al., 2023; Tang et al., 2024; Gong et al., 2024; Deshmukh et al., 2023; Kong et al., 2024; Chen et al., 2024a] typically uses an encoder to process input signals, where the encoder may be pre-trained on unpaired speech

or audio data, or trained on downstream tasks such as ASR. The encoder's output is then processed by an optional adaptor and fed into the LLM. These models specify tasks through text instruction prompts and have demonstrated strong performance on various tasks.

The second category of models [Défossez et al., 2024; Fang et al., 2024; Xie and Wu, 2024] focuses on seamless voice interaction. These models usually utilize speech tokenization techniques to represent and process speech signals, so that speech and text data can be treated as similar types of data for training models. However, these models rarely focus on understanding audio information.

In contrast to these approaches, SOLLA distinguishes itself by emphasizing both the understanding of audio content and the processing of speech instructions. It is specifically designed to generate appropriate responses by leveraging this dual understanding.

Benchmarks for Speech/Audio LLMs The growing interest in speech/audio LLMs has led to the development of new benchmark datasets [Chen et al., 2024c; Wang et al., 2024a; Huang et al., 2024b,a; Ao et al., 2024; Sakshi et al., 2024; Yang et al., 2024b; Chen et al., 2024b] aimed at evaluating these models from various perspectives. Many of these benchmarks [Wang et al., 2024a; Yang et al., 2024b; Sakshi et al., 2024; Huang et al., 2024b,a; Chen et al., 2024b] focus on testing models' performance only based on text instructions.

In contrast, VoiceBench [Chen et al., 2024c] evaluates system performance through both synthetic and real spoken instructions. This benchmark assesses general knowledge, instruction-following ability, and safety compliance across diverse speaker and environmental conditions, with audio containing background sounds unrelated to the speech instruction itself. This design serves to test the model's robustness to noise.

Similarly, SD-Eval [Ao et al., 2024] measures spoken dialogue models across mul-

tiple dimensions, such as linguistic content, paralinguistic cues (emotion, accent, age), and environmental context, through four sub-tasks. The *test-env* subset of SD-Eval, which contains 690 sentences, includes audio and speech instructions. Here, the model must use information from the audio to provide answers, though the focus is on daily conversations.

Unlike the above benchmarks, SA-Eval is specifically designed for audio-related tasks such as audio classification, captioning, and question answering. It creates specific speech instructions based on these tasks and categorizes the difficulty into *easy* mode and *hard* mode according to real-life situations.

4.2.3 SOLLA Framework

SOLLA deals with speech instructions and audio context understanding. We formulate the task as follows: given a speech mixture

$$s_m = s_s + s_a \tag{4.1}$$

where s_s represents the speech instruction and s_a denotes the accompanying audio signal. The model must generate a text response that integrates both types of information. For example, the speech instruction may prompt the model to predict an event occurring in the audio or ask a related question. The subsequent sections detail each module.

4.2.3.1 Overview

SOLLA is a speech-oriented large language model. The framework of SOLLA is illustrated in Figure 4.20. The SOLLA is built upon four key components: an audio encoder, an adaptor, an Audio Tagging (AT) module, and a large language model (LLM). In addition to generating the corresponding answer, SOLLA also has an additional ASR-

assisted prediction task to aid in understanding speech instructions.

4.2.3.2 Audio Encoder

The model's audio encoder is the encoder part of Whisper-Large-v3 [Radford et al., 2023]. We choose the Whisper encoder as the audio encoder for two reasons. First, whisper is trained on 680,000 hours of multilingual data with multitask supervision, which yields high accuracy and robustness in speech recognition. Consequently, its encoder is widely adopted for processing speech data [Chu et al., 2023, 2024; Tang et al., 2024]. Second, recent work [Gong et al., 2023a] also demonstrates that its representations are noise-variant, capturing rich audio information. Accordingly, we use the Whisper encoder to efficiently extract speech and audio information from the input speech mixture s_m. During training, we freeze the encoder parameters and fine-tune the audio encoder using a trainable LoRA adaptor [Hu et al., 2021].

4.2.3.3 Adaptor

The adaptor is designed to transform the raw output from the audio encoder into a suitable format that the LLM can process effectively. Specifically, it comprises two linear layers: the first is followed by a GELU activation function [Hendrycks and Gimpel, 2016], and the second is succeeded by a two-dimensional average pooling operation that downsamples the representation by a factor of two. The final output is a sequence representation $\mathbf{v_a} = (v_a^1, v_a^2, ..., v_a^n)$, where n denotes its length.

4.2.3.4 AT Module

To capture audio-specific information, we introduce a novel Audio Tagging (AT) module. This module begins by applying a two-dimensional average pooling operation to the audio encoder outputs, reducing the temporal resolution by a factor of twenty. The pooled features are then normalized via layer normalization and transformed

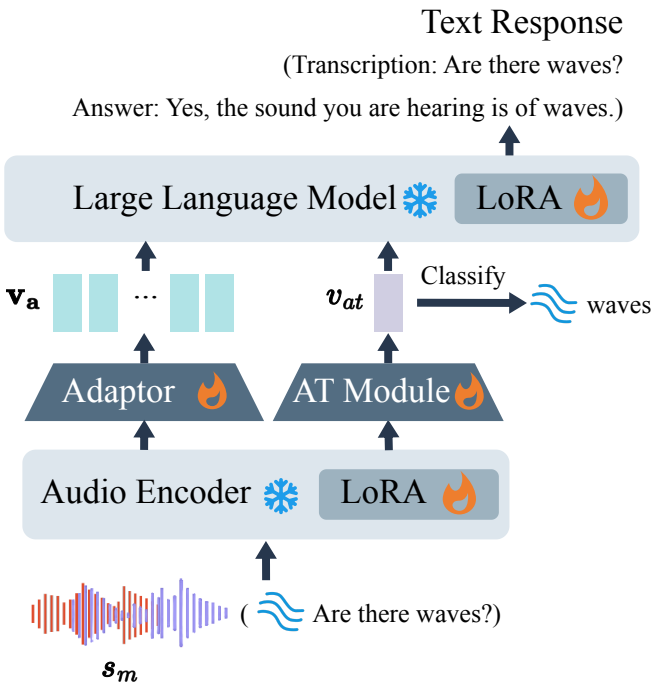


Figure 4.20: The overview framework of SOLLA. AT module denotes the audio tagging module.

through a linear projection to yield an intermediate representation. A residual attention block subsequently models long-range temporal dependencies. After mean-pooling along the temporal dimension, the resulting compact feature vector v_{at} is normalized and passed through a linear classifier to produce logits for audio event prediction.

4.2.3.5 Large Language Model

We employ the InternLM2-chat model (InternLM2-chat-7b) [Cai et al., 2024], a 7-billion-parameter large language model optimized for dialogue generation. During training, the LLM remains frozen while a trainable LoRA adaptor [Hu et al., 2021] is employed to enable fine-tuning. The LLM takes the adaptor's output representation $(v_a^1, v_a^2, ..., v_a^n)$ and the AT module's pooled vector v_{at} as its inputs.

4.2.3.6 ASR-Assisted Prediction

To enhance the model's comprehension of speech instructions, we incorporate an ASR-assisted prediction strategy. During training, the model first performs the ASR task before generating the final answer. For example, the target output may be *Transcription: Are there waves? Answer: Yes, the sound you are hearing is of waves*, as shown in Figure 4.20. This approach ensures that the model accurately captures the content of the speech instruction, which is then combined with the audio information to produce the final response.

4.2.4 SA-Eval Benchmark Dataset

4.2.4.1 Overview

The SA-Eval dataset is derived from test sets of widely used datasets spanning three different tasks, as shown in Table 4.7. To ensure diversity in both question formulation and vocal characteristics of speech instructions, we leverage GPT-4o [OpenAI, 2024] to generate multiple question formats for each task while employing TTS models to synthesize varied speaking styles. Below, we provide a detailed explanation of the dataset construction process.

Table 4.7: Statistics of the SA-Eval benchmark dataset, which is built upon a few open-source datasets for audio classification, audio captioning and audio QA. We reuse them and mix them with speech instructions for SA-Eval. In addition, we divide the SA-Eval into *easy* and *hard* subsets based on SNR.

Dataset	Task	Average Duration (s)	# QA Pairs	# Instructions	SNR	
					easy	hard
VGGSound [Chen et al., 2020a]	Audio Classification	12.4	15,341	402	-0.8	-10.0
AudioSet [Gemmeke et al., 2017]	Audio Classification	12.3	17,241	402	-2.3	-11.8
FSD50K [Fonseca et al., 2021]	Audio Classification	12.6	10,231	402	-0.9	-12.6
AudioCaps [Kim et al., 2019]	Audio Captioning	12.2	881	345	-1.8	-11.5
Clotho V2 [Drossos et al., 2020]	Audio Captioning	24.3	1,045	355	-6.1	-14.4
Clotho-AQA [Lipping et al., 2022]	Audio Question Answering	23.6	8,430	2515	-9.7	-17.4

4.2.4.2 Base Datasets

We construct the SA-Eval dataset based on the test sets of six widely used datasets, including VGGSound [Chen et al., 2020a], AudioSet [Gemmeke et al., 2017], FSD50K [Fonseca et al., 2021], AudioCaps [Kim et al., 2019], Clotho V2 [Drossos et al., 2020] and Clotho-AQA [Lipping et al., 2022], encompassing three different tasks, audio classification, audio captioning and audio question answering. Please note that several test sets, including VGGSound, AudioSet, and AudioCaps, are missing some audio files due to data corruption or unavailability of the source files.

4.2.4.3 Generating Text Instructions

Except for Clotho-AQA, which includes questions corresponding to each audio, we generate diverse instructions for the other two tasks. Specifically, we begin by formulating multiple instructions for these tasks and then use a rewriting prompt with GPT-4o [OpenAI, 2024] to produce 100 unique sentences for each instruction. Upon reviewing the outputs, we find that some generated instructions are ill-suited for the target tasks. As a result, we engage human annotators to filter out any unsuitable instructions. Details can be found in Section 4.2.7.1. Table 4.7 shows how many unique instructions each test set contains finally.

4.2.4.4 Generating Speech Instructions

We generate speech for our instructions by leveraging MaskGCT [Wang et al., 2024e] together with Microsoft Azure TTS [11]. Specifically, we select speakers from the Emilia dataset [He et al., 2024] who have at least seven speech recordings. We then retain the top seven recordings ranked by DNSMOS scores [Reddy et al., 2021]. This process yields a pool of 9,880 speakers and 69,160 audio recordings that are randomly used as

[11] https://learn.microsoft.com/en-us/azure/ai-services/speech-service/text-to-speech

speech prompts for MaskGCT.

Following speech generation with MaskGCT, we observe that a small fraction of the speech exhibits issues, such as excessive length, content inaccuracies, or language errors. To address this, we further filter these problematic recordings based on the content and regenerate this subset of speech using Microsoft Azure TTS. Specifically, we first calculate the word error rate (WER) for each utterance generated by MaskGCT. Those with a WER exceeding 40% are regenerated, while human annotators manually review utterances with lower WERs to determine if regeneration is necessary. The speaker types for Microsoft Azure TTS are randomly selected.

4.2.4.5 Mixing Audio Events and Speech Instructions

In real-world scenarios, speech instructions and accompanying audio can appear in various configurations. For instance, the speech instruction might precede the audio, or the two might overlap. To simulate these diverse conditions and evaluate model performance across different difficulty levels, we generate test sets in two modes, *easy* and *hard*, as detailed in Algorithm 2 of Section 4.2.7.2. In our algorithm, the *easy* mode features non-overlapping audio with a higher SNR, whereas the *hard* mode involves overlapping audio with a lower SNR, thereby mimicking challenging real-world situations where high audio volume can obscure speech instructions.

To create these datasets, we first decide whether to overlap the audio signal s_a and the speech instruction s_s, applying appropriate padding when necessary. In the *hard* mode, s_a and s_s always overlap, while in the *easy* mode, they remain non-overlapping. Second, we sample the loudness of both s_a and s_s using loudness units relative to full scale (LUFS) [Series, 2011], following the mixing strategy of LibriMix [Cosentino et al., 2020]. LUFS, defined by the ITU-R BS.1770-4 recommendation [Series, 2011], better reflects perceived loudness than traditional SNR metrics, as it is unaffected by silence and remains relatively insensitive to downsampling [Cosentino et al., 2020].

For the *easy* mode, both signals are uniformly sampled between -30 and -25 LUFS. In the *hard* mode, s_a is sampled between -23 and -20 LUFS, while s_s is sampled between -33 and -30 LUFS to simulate scenarios where the speech instruction is less clear. When necessary, the signals are clipped to 0.9 before being mixed to produce the final mixture s_m.

4.2.5 Experimental Setups

4.2.5.1 Training Set Preparation

We prepare the training set using a procedure similar to that of the SA-Eval dataset. Specifically, we construct the set using the training data from FSD50K [Fonseca et al., 2021], AudioSet[Gemmeke et al., 2017], AudioCaps [Kim et al., 2019], FreeSound[Font et al., 2013], Sound Bible [SoundBible, 2006], VGGSound [Chen et al., 2020a], Clotho V2 [Drossos et al., 2020], Clotho-AQA [Lipping et al., 2022], and MusicQA [Liu et al., 2024b]. For all datasets except Clotho-AQA and MusicQA, we use the instructions from OpenAQA [Gong et al., 2024]. There are several differences compared to the preparation of the SA-Eval dataset.

First, we generate the corresponding speech instructions using MaskGCT [Wang et al., 2024e] without additional filtering. Second, the parameters for speech instruction and audio mixing are different: 80% of the speech instructions and audio overlapped, and we randomly set the loudness of speech instructions between -38 and -20 LUFS, while adjusting the audio loudness between -25 and -33 LUFS. Finally, we replace 20% of the speech instructions with those from SA-Eval for audio captioning and audio classification tasks. Detailed data statistics of the training set are provided in Section 4.2.7.3.

4.2.5.2 Configurations for Model Training

All implemented models are developed using Xtuner [Contributors, 2023]. The optimization is performed with the AdamW optimizer [Loshchilov and Hutter, 2017], employing a learning rate of 2×10^{-4}. Model fine-tuning is conducted over two epochs on 16 A100 GPUs, each processing a batch size of 16, with each experiment taking approximately four days. For the LoRA adapter of the InternLM2-chat-7b model, we set the rank to 512 and α to 256. In contrast, for the encoder of the Whisper large-v3 model, the rank is set to 64 and α to 16.

4.2.5.3 Baseline Models

In addition to SOLLA, we implement two baseline models for comparison, namely Audio LLM_{text} and Audio $\text{LLM}_{\text{speech}}$. Unlike SOLLA, these two models consist only of an audio encoder, an adaptor, and an LLM. They use the answer as the target without ASR-assisted prediction. All other training parameters remain the same as SOLLA.

The first baseline, Audio LLM_{text}, takes audio and text instruction as input. This model allows us to assess the effect of using a speech versus a text instruction prompt on performance. In contrast, Audio $\text{LLM}_{\text{speech}}$ receives the same input as SOLLA.

4.2.5.4 Publicly Available Models

We compare two types of models: one guided by text instructions and the other by speech instructions. For models using text instruction, we strictly follow the original studies' evaluation protocols to ensure fairness, focusing only on the in-distribution test sets that correspond to their training data. For models that do not report their training sets, including Qwen-Audio [Chu et al., 2023] and Qwen2-Audio [Chu et al., 2024], we selectively compare the test sets or tasks evaluated in their papers. In contrast, for models using speech instruction, we employ SA-Eval for evaluation, as detailed in Section 4.2.4. Overall, our analysis includes the following publicly available

models for comparison:

Pengi [Deshmukh et al., 2023] Pengi reformulates both open- and closed-ended audio tasks as text-generation problems. It uses transfer learning and instruction-tuned templates to process audio and text without task-specific fine-tuning.

Audio Flamingo [Kong et al., 2024] Audio Flamingo extends large language models to interpret diverse audio, including non-speech sounds. It excels in audio comprehension, few-shot learning, and multi-turn dialogues.

LTU [Gong et al., 2024] LTU fuses audio perception with reasoning. Trained on the OpenAQA-5M dataset using a perception-to-understanding curriculum, it outperforms rivals in audio classification and captioning tasks.

Qwen-Audio [Chu et al., 2024] Qwen-Audio is pre-trained on over 30 tasks, including speech, ambient sounds, and music, in multiple languages. Its broad training yields strong performance without specialized fine-tuning.

Qwen2-Audio [Chu et al., 2024] Qwen2-Audio builds on Qwen-Audio, using larger-scale training data and Direct Preference Optimization [Rafailov et al., 2024] for improved alignment. It offers two modes: Voice Chat (Qwen2-Audio-VC) for dialogue using speech instruction and Audio Analysis (Qwen2-Audio-AA) for detailed processing using text instruction.

GPT-4o [OpenAI, 2024] GPT-4o [12] combines advanced language understanding with enhanced speech recognition and generation, supporting tasks from creative writing to real-time conversation.

[12] The version of GPT-4o is `gpt-4o-audio-preview-2024-12-17`.

4.2.5.5 Evaluation Metrics and Methods

For single-label audio classification and audio question answering, we gauge performance with accuracy (%). We report two results on the Clotho-AQA test set. The first, ACC_{All}, represents the overall accuracy across all test data. The second, ACC_{Una}, measures the accuracy on questions where the answers from all three annotators are consistent, with "Una" standing for unanimous. For multi-label audio classification, we adopt the F1 score, applying the macro averaging approach. For audio captioning, we rely on CIDEr [Vedantam et al., 2015], SPICE [Anderson et al., 2016], and SPIDEr [Liu et al., 2017].

For single-label audio classification and audio question answering tasks, if the model's recognized answer exactly matches the ground truth, we consider it correct. Otherwise, we rely on o1-mini[13] to assess whether the output is correct. The specific prompts can be found in Section 4.2.7.4. For multi-label audio classification, if the model's predicted label does not match any of the known labels, we calculate the cosine similarity between the text embedding [14] of the predicted label and those of all the known labels, and select the label with the highest similarity as the predicted label.

4.2.6 Experimental Results and Analysis

4.2.6.1 Main Results

Table 4.8 shows the results of both publicly available models and our self-implemented models across six test sets of three tasks. In all cases, higher metric values correspond to better performance.

First, the experimental results show that SOLLA performs on par with or exceeds Audio $\text{LLM}_{\text{speech}}$ on SA-Eval. In particular, SOLLA exhibits significantly better results

[13] The version of o1-mini is `o1-mini-2024-09-12`.
[14] The model for extracting text embedding is `text-embedding-3-large`.

Table 4.8: Main results of publicly available models and self-implemented models across six test sets of three tasks. For all metrics, larger numbers indicate better performance. The dash indicates that the model is not tested on the relevant dataset in the original paper. † Pengi's AudioCaps results come from the original paper. Due to data corruption or unavailability of the source files, the test data used may be different from that used by other models, as explained in Section 4.2.4.2.

Model	Audio Classification			Audio Captioning		Audio Question Answering
	VGGSound ACC	AudioSet F1	FSD50K F1	Clotho V2 CIDEr / SPICE / SPIDEr	AudioCaps CIDEr / SPICE / SPIDEr	Clotho-AQA ACC_{All} / ACC_{Una}
Models using Text Instruction						
Pengi † [Deshmukh et al., 2023]	-	-	52.6	41.6 / 12.6 / 27.1	75.2 / 18.2 / 46.7	46.4 / 62.3
Audio Flamingo [Kong et al., 2024]	-	-	56.0	46.5 / 12.8 / 29.8	- / - / -	58.5 / 86.9
LTU [Gong et al., 2024]	53.6	28.3	47.8	31.6 / 12.0 / 21.8	48.5 / 16.8 / 32.5	- / -
Qwen-Audio [Chu et al., 2023]	-	-	-	44.1 / 13.6 / 28.8	- / - / -	57.9 / 74.9
Qwen2-Audio-AA [Chu et al., 2024]	-	-	-	32.8 / 10.8 / 21.8	47.9 / 16.5 / 32.2	- / -
Audio LLM_{text}	55.6	26.0	57.4	39.4 / 12.5 / 25.9	62.7 / 15.6 / 39.2	62.2 / 87.9
SA-Eval - easy mode						
Qwen2-Audio-VC [Chu et al., 2024]	10.9	-	-	- / - / -	- / - / -	45.7 / 59.2
GPT-4o [OpenAI, 2024]	7.7	-	-	- / - / -	- / - / -	45.8 / 56.9
Audio LLM_{speech}	53.4	25.2	53.1	38.7 / 12.4 / 25.6	64.0 / 15.5 / 39.8	61.7 / 86.8
Solla	53.8	25.2	55.1	38.5 / 12.6 / 25.6	63.2 / 16.0 / 39.6	62.1 / 86.9
SA-Eval - hard mode						
Qwen2-Audio-VC [Chu et al., 2024]	18.3	-	-	- / - / -	- / - / -	34.3 / 46.2
GPT-4o [OpenAI, 2024]	4.8	-	-	- / - / -	- / - / -	35.2 / 39.1
Audio LLM_{speech}	53.7	23.7	51.6	37.2 / 12.1 / 24.6	58.8 / 14.5 / 36.7	54.8 / 75.7
Solla	53.5	23.8	54.0	38.6 / 12.3 / 25.5	60.6 / 15.4 / 38.0	55.4 / 78.3

than Audio LLM_{speech} in *hard* mode across most test sets. This highlights the robustness of Solla across various scenarios.

Second, Audio LLM_{text} provides a solid baseline for comparison as it outperforms other publicly available models on VGGSound, FSD-50K, and Clotho-AQA. Its performance drops on other test sets, possibly due to the training data. As detailed in Section 4.2.5.1, the majority of its training instructions come from OpenAQA [Gong et al., 2024], which mainly offers open-ended instructions. Consequently, Audio LLM_{text} also outperforms LTU which is trained using OpenAQA across nearly all test sets.

Third, comparing the results of Solla and Audio LLM_{text} reveals that Solla's performance has declined across all test sets, with a more pronounced drop in *hard* mode. This indicates that even with the assistance of the AT Module and ASR-assisted prediction, the model still has room for improvement in understanding speech instructions and audio information.

Finally, we compare Solla with two models using speech instruction, Qwen2-

Audio-VC and GPT-4o. We focus our evaluation on VGGSound and Clotho-AQA for two reasons. First, the metrics used for audio captioning do not account for synonyms [Gong et al., 2024], so comparing results for models not trained on the relevant datasets would be unfair. Second, as single-label audio classification and audio question answering datasets, VGGSound and Clotho-AQA better reflect a model's ability to recognize audio events and comprehend audio information, unlike multi-label audio classification requiring multiple audio event outputs and more complicated evaluation. Experimental results show that SOLLA significantly outperforms GPT-4o and Qwen2-Audio-VC in both evaluation modes.

Table 4.9: Ablation Study for AT Module and ASR-assisted prediction on SA-Eval. "w/o" refers to without. For all metrics, larger numbers indicate better performance.

Model	Audio Classification			Audio Captioning		Audio Question Answering
	VGGSound	AudioSet	FSD50K	Clotho V2	AudioCaps	Clotho-AQA
	ACC	F1	F1	CIDEr / SPICE / SPIDEr	CIDEr / SPICE / SPIDEr	ACC_{All} / ACC_{Una}
SA-Eval - easy mode						
SOLLA	53.8	25.2	55.1	38.5 / 12.6 / 25.6	63.2 / 16.0 / 39.6	62.1 / 86.9
w/o AT Module	53.1	24.4	54.1	38.4 / 12.8 / 25.6	64.2 / 16.3 / 40.3	61.5 / 85.8
w/o ASR-Assisted Pred	54.1	24.4	53.3	40.0 / 12.5 / 26.3	62.8 / 15.5 / 39.2	61.4 / 86.3
w/o Both (Audio LLM_{speech})	53.4	25.2	53.1	38.7 / 12.4 / 25.6	64.0 / 15.5 / 39.8	61.7 / 86.8
SA-Eval - hard mode						
SOLLA	53.5	23.8	54.0	38.6 / 12.3 / 25.5	60.6 / 15.4 / 38.0	55.4 / 78.3
w/o AT Module	53.0	23.1	52.9	38.1 / 12.4 / 25.2	61.6 / 15.7 / 38.7	54.8 / 76.4
w/o ASR-Assisted Pred	54.1	23.3	51.8	37.5 / 12.1 / 24.8	58.7 / 14.7 / 36.7	55.6 / 76.9
w/o Both (Audio LLM_{speech})	53.7	23.7	51.6	37.2 / 12.1 / 24.6	58.8 / 14.5 / 36.7	54.8 / 75.7

4.2.6.2 Ablation Study

We conduct ablation studies on the SA-Eval to assess the impact of the AT module and ASR-assisted prediction on performance, as shown in Table 4.9.

First, on the *easy* mode of SA-Eval, we observe similar performance across various models. This can be attributed to the non-overlap data in this mode, which leads to similar model behaviour when processing speech instructions and audio information. The analysis in Section 4.2.6.3 further supports this point.

Second, our experimental results show that incorporating the AT Module enhances model performance compared to models without it on audio classification

and audio question answering tasks. Specifically, models with the AT Module, such as SOLLA and SOLLA without ASR-assisted prediction, outperform their counterparts lacking the module (i.e., SOLLA without AT Module and SOLLA without both) across most datasets. For the audio captioning task, performance variations may arise from the fact that recognizing audio events does not directly influence captioning results. Captioning performance is also influenced by factors such as the model's ability to summarize and generalize.

Finally, models utilizing ASR-assisted prediction show improved performance relative to those without it on most of the test sets, with particularly notable gains observed in the audio captioning task. These findings highlight the benefits of integrating both the AT Module and ASR-assisted prediction into our model architecture.

Table 4.10: Results for ASR-assisted prediction and instruction-following accuracy. "IF" stands for instruction following, "Pred" stands for prediction, and "w/o" refers to without.

Model	**Mode of SA-Eval**	**ASR** WER ↓	**IF** ACC ↑
SOLLA	*easy*	0.0	100.0
w/o ASR-Assisted Pred		-	100.0
SOLLA	*hard*	1.9	98.9
w/o ASR-Assisted Pred		-	98.3

4.2.6.3 Analysis of ASR-Assisted Prediction and Instruction-Following Ability

To better understand the role of ASR-assisted prediction, we further analyze the ASR-assisted prediction results and the model's instruction-following ability. We conduct analysis on the FSD50K test set using WER and accuracy metrics, as shown in Table 4.10.

In the *easy* mode, the ASR achieves a WER of zero and the instruction-following

accuracy reaches 100%, suggesting that the performance gap between SOLLA and Audio LLM_{text} also stems from differences in audio comprehension, with speech instructions potentially influencing the model's understanding of audio inputs.

In contrast, in the *hard* mode, SOLLA outperforms SOLLA w/o ASR-assisted prediction, showing higher instruction-following accuracy. This highlights that ASR-assisted prediction enhances the model's ability to interpret speech instructions.

4.2.7 Supplementary Details

4.2.7.1 Instruction Generation Details

First, we formulate five different base instructions for the audio captioning and audio classification tasks. The content can be summarized by completing the specific task based on background sound. Subsequently, we utilize GPT-4o [OpenAI, 2024] to paraphrase each instruction by 100 times, using the prompt: *Generate 100 new versions of this sentence with synonyms: {base instruction}.*

As discussed in Section 4.2.4.3, the generated instructions contain overly specific words that cannot be universally applied to all test cases. For example, certain terms, such as *harmonics* for background sound, are used in contexts where they do not apply. Therefore, human annotators, who are the student volunteers, identify these words and delete the instructions that contain them. Table 4.11 shows some examples of instructions for the audio classification and audio captioning tasks.

4.2.7.2 Mixing Algorithm of Audio and Speech Instruction

Algorithm 2 shows the detailed steps of mixing audio and speech instruction.

Table 4.11: Examples of instructions for the audio classification and audio captioning tasks.

Task	Example
Audio Classification	Classify all the sounds in the background and let me know their labels. Can you identify and provide the sound labels for all events from the background? Please identify every audio event in the background and provide its label. Could you tell me the names of the sound labels for all audio events in the background? Could you identify and classify the audio events present in the background sound?
Audio Captioning	Can you summarize the audio in the background with a caption? Would you draft a caption describing the ambient sound and share it with me? Could you create a caption that describes the background audio and let me know? Write a description of the background audio. Describe the background audio in a caption, and share it with me.

4.2.7.3 Training Set Statistics

Table 4.12 presents the statistics of the training data. Except for MusicQA [Liu et al., 2024b] and ClothoAQA [Lipping et al., 2022], which use the original question as the instruction, the rest are sourced from the OpenAQA dataset [Gong et al., 2024].

4.2.7.4 Evaluation Judgment

For single-label tasks, VGGSound and Clotho-AQA, an English normalizer is first applied to both the ground-truth and prediction outputs. The outputs are considered a correct answer if they are found to be identical by character. Otherwise, the o1-mini is utilized to evaluate the prediction's correctness, with prompts shown in Figure 4.21 and Figure 4.22.

```
Given a question, you are tasked with evaluating whether the AI assistant's
answer is correct.

1. Input: You will be given the following:
  • A question.
  • The answer from the AI assistant.
  • The ground truth answer to the question.
2. Evaluation: Compare the intended answer with the ground truth and determine
   its correctness.
  • If the answer of the ground truth is mentioned / included in AI
    assistant's answer, output "correct".
  • Otherwise, output "wrong".
3. Explanation:
  • After making your judgment, provide an explanation for your decision.
  • The explanation should clearly explain why the answer is either correct or
    wrong based on the comparison with the ground truth.
4. Output Format:
  • The final output should consist of two parts:
    I. Judgment: Either "correct" or "wrong".
    II.Explanation: A brief explanation for your judgment.
  • These two parts should be separated by a hyphen (`judgment-explanation`).

Input for you to process:
[Question]:
{transcript}
[Answer]:
{output}
[Ground Truth]
{target}
```

Figure 4.21: Evaluation judgment prompt of VGGSound.

```
Given a question, you are tasked with evaluating whether the AI assistant's
answer is correct.

1. Input: You will be given the following:
  • A question.
  • The answer from the AI assistant.
  • The ground truth answer to the question.
2. Evaluation: Compare the intended answer with the ground truth and determine
   its correctness.
  • If the answer of the AI assistant aligns with the ground truth, output
    "correct".
  • If the answer of the AI assistant does not align with the ground truth,
    output "wrong".
3. Explanation:
  • After making your judgment, provide an explanation for your decision.
  • The explanation should clearly explain why the answer is either correct or
    wrong based on the comparison with the ground truth.
4. Output Format:
  • The final output should consist of two parts:
    I. Judgment: Either "correct" or "wrong".
    II.Explanation: A brief explanation for your judgment.
  • These two parts should be separated by a hyphen (`judgment-explanation`).

Input for you to process:
[Question]:
{transcript}
[Answer]:
{output}
[Ground Truth]
{target}
```

Figure 4.22: Evaluation judgment prompt of Clotho-AQA.

4.2.8 Summary

This chapter introduced SOLLA, a speech-oriented LLM that hears acoustic context. SOLLA combines the AT module and ASR-assisted prediction to effectively interpret audio cues and accurately understand spoken questions. In addition, we presented SA-Eval, a comprehensive benchmark designed to evaluate models on six test sets for audio classification, audio captioning, and audio question answering, across two levels of difficulty.

Our experimental results demonstrate that SOLLA performs on par with or outperforms baseline models, particularly in the *hard* mode, highlighting its robustness across different scenarios. Furthermore, the analysis reveals that the ASR-assisted prediction method enhances the model's ability to interpret speech instructions, while the AT module may play a crucial role in understanding the audio content of the input signal.

The limitations of SOLLA and SA-Eval are as follows: First, SOLLA and SA-Eval focus on scenarios where the input is audio and the output is text, thereby restricting the output to text form. Second, SOLLA and SA-Eval only address understanding tasks in the audio events, without considering music-related aspects. Lastly, SA-Eval's evaluation currently focuses only on single-turn dialogues, limiting its ability to assess multi-turn complex dialogues.

□ End of chapter.

Algorithm 2: Generating Mixture of Audio and Speech Instruction

Data:
s_s: Input signals of the speech instruction
s_a: Input signals of the audio
l_a: Length of the audio
l_s: Length of the speech instruction
m: *easy* or *hard*

if *m is hard* **then**
  `// If overlap, selecting start time and padding the shorter one`
  if $l_a < l_s$ **then**
    $t_s \leftarrow randint(0, l_s - l_a)$;
    $s_a \leftarrow pad(s_a, (t_s, l_s - l_a - t_s))$;
  else
    $t_s \leftarrow randint(0, l_a - l_s)$;
    $s_s \leftarrow pad(s_s, (t_s, l_a - l_s - t_s))$;
  end
else if *m is easy* **then**
  `// If not overlap, padding both audio and speech`
  if *random()* > 0.5 **then**
    $s_a \leftarrow pad(s_a, (l_s, 0))$;
    $s_s \leftarrow pad(s_a, (0, l_a))$;
  else
    $s_a \leftarrow pad(s_a, (0, l_s))$;
    $s_s \leftarrow pad(s_a, (l_a, 0))$;
  end
`// Selecting loudness for speech instruction and audio`
if *m is hard* **then**
  $v_a \leftarrow randint(-23, -20)$;
  $v_s \leftarrow randint(-33, -30)$;
else if *m is easy* **then**
  $v_a \leftarrow randint(-38, -20)$;
  $v_s \leftarrow randint(-33, -25)$;
`// Setting loudness and clipping if needed`
$s_a \leftarrow set\_loudness(s_a, v_a)$;
$s_s \leftarrow set\_loudness(s_s, v_s)$;
`// Mixing and getting mixture speech` s_m
$s_m \leftarrow s_s + s_a$;
Output: speech mixture s_m

Table 4.12: Statistics of the training set.

Dataset	# QA Pairs	# Audio Samples	Duration (hrs)
FSD50K [Fonseca et al., 2021]	441,119	40,128	79.8
AudioSet-2M [Gemmeke et al., 2017]	394,865	394,865	1087.3
AudioSet-20K [Gemmeke et al., 2017]	194,970	18,301	50.4
AudioSet Strong [Hershey et al., 2021]	1,276,793	99,865	273.7
AudioCaps [Kim et al., 2019]	491,326	45,639	125.0
FreeSound [Font et al., 2013]	768,822	86,845	107.8
Sound Bible [SoundBible, 2006]	16,225	1,104	2.2
VGGSound [Chen et al., 2020a]	1,104,020	181,700	503.7
Clotho V2 [Drossos et al., 2020]	117,406	4,780	29.9
MusicQA [Liu et al., 2024b]	104,106	12,542	76.2
ClothoAQA [Lipping et al., 2022]	5,666	1,174	7.4
Overall	4,915,318	770,351	2,023.6

Chapter 5

Spoken Language Models that Think Aloud

Summary

This chapter introduces a Thinker-Talker framework featuring a novel think-aloud module inspired by human speech production. The system orchestrates two asynchronous streams: a reasoning stream for logical deduction and a speaking stream for continuous, incremental think-aloud response generation. A dynamic balance strategy ensures alignment between these streams, substantially reducing unmasked silence latency while maintaining reasoning performance comparable to serial reasoning baselines.

If Chapter 4 studies how spoken language models can perceive information beyond lexical content, then the next question is whether such systems can also reason and interact naturally in speech form. In many spoken applications, especially those involving complex question answering or multi-step inference, strong reasoning ability

is essential. However, directly applying the conventional think-then-speak paradigm often leads to long silent delays, which significantly degrade the responsiveness and naturalness of spoken interaction.

Against this background, this chapter moves from beyond-word perception to spoken reasoning and interaction. It investigates how spoken language models can maintain strong reasoning ability while reducing the perceived latency experienced by users. To this end, the chapter presents a think-aloud framework in which reasoning and speaking proceed asynchronously, allowing the system to expose intermediate verbalization while continuing its internal reasoning process. As the final technical chapter of the dissertation, this chapter extends the overall progression from speech representation learning, to robustness, to perception, and ultimately to low-latency cognitive spoken interaction.

5.1 Introduction

Human speakers rarely plan an entire response silently before delivering it [Levelt, 1993]. Instead, they think while speaking; ideas are formulated, revised, and refined within the flow of conversation. Disfluencies, such as pauses, hedges, and self-corrections, facilitate turn-taking and signal confidence, progress, and intent in real time [Clark and Fox Tree, 2002]. Content is organized incrementally: speakers typically provide a basic structure before details, adjusting mid-utterance as new information arises or in response to listener feedback. This interactive mode of production ensures naturalness and maintains conversational synchrony, even while the underlying reasoning is still unfolding.

The integration of complex reasoning has become a cornerstone of recent LLM research [Team, 2025; Guo et al., 2025; Jaech et al., 2024; Wei et al., 2022], driving significant performance gains in mathematical problem-solving and code generation

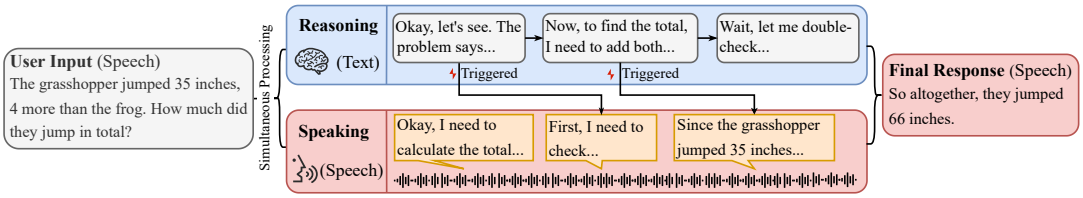


Figure 5.1: Illustration of the asynchronous reasoning and think-aloud generation mechanism. In this example, two think-aloud responses are triggered during the reasoning generation phase. Notably, the first think-aloud response is forcibly triggered to ensure it is generated synchronously with the beginning of reasoning. Finally, the model generates the final response based on both the reasoning text and the think-aloud responses.

[Guo et al., 2025; Shao et al., 2025; Jaech et al., 2024]. By enabling models to decompose abstract queries into logical, multi-step deductions, reasoning capabilities have transformed LLMs from simple text generators into robust engines for algorithmic execution. This shift emphasizes the critical role of reasoning in advancing general-purpose AI.

SLMs have recently garnered significant attention [Zeng et al., 2024; Li et al., 2025b; Xu et al., 2025; Chu et al., 2024; Fang et al., 2025]. An intuitive approach to incorporating reasoning into these systems is to perform full textual deliberation before speaking, followed by generating the final response. However, this serial 'think-then-speak' design is often incompatible with spoken conversation. Since users are accustomed to immediate feedback, the resulting delays disrupt natural turn-taking and make the interaction feel unresponsive. In practice, placing explicit reasoning before speech introduces significant latency, rendering such systems impractical for fluid interactive scenarios.

The concept of "thinking aloud" originates from cognitive psychology and protocol analysis, established as a method to investigate human cognitive processes [Ericsson and Simon, 1984] and grounded in the information processing theory of problem solving [Newell and Simon, 1972]. Inspired by this mechanism, we redefine think-

aloud responses in SLMs as the generation of intermediate speech during the model's reasoning phase.

As illustrated in Figure 5.1, our proposed paradigm externalizes the reasoning content by initiating simultaneous processing upon receiving user input: a reasoning stream performs textual deduction, while a speaking stream converts these internal states into audible think-aloud responses. Specifically, the system prioritizes responsiveness by immediately generating an initial think-aloud utterance (e.g., a conversational filler or query paraphrase) to bridge the start-up gap. Subsequently, as reasoning progresses, intermediate milestones trigger additional speech segments, effectively filling long silent gaps and providing earlier audible feedback before the final response is ready.

To enable SLMs to speak and think concurrently, we introduce a framework based on the Thinker-Talker architecture, featuring a novel think-aloud module that allows the model to think while speaking. Our system orchestrates two asynchronous token streams, reasoning and speaking, to ensure fluid parallel generation. In summary, our main contributions are as follows:

- We implement a Thinker-Talker architecture featuring a think-aloud module that orchestrates two asynchronous streams. This mechanism enables the model to process complex logic (reasoning stream) and generate think-aloud responses (speaking stream) simultaneously, rather than waiting for the reasoning to conclude.

- We design a dynamic balance strategy to coordinate the asynchronous streams. By adaptively regulating the generation budget, this approach seeks to reduce latency and support fluid speech that stays synchronized with the reasoning process.

- Experimental results suggest that our approach substantially reduces unmasked

silence latency while maintaining reasoning performance comparable to serial reasoning baselines, facilitating answers that are both logically grounded and temporally consistent.

5.2 Related Work

5.2.1 Spoken Language Models

This section categorizes the speech generation mechanisms of existing SLMs into two primary paradigms: (1) Interleaved text-speech decoding [Zeng et al., 2024; Li et al., 2025b] and (2) the Thinker-Talker architecture [Xu et al., 2025; Ding et al., 2025; Fang et al., 2025]. In the interleaved paradigm, the SLM backbone alternately generates chunks of text and speech tokens. Crucially, the text tokens serve as a direct transcription, guiding the content of the immediate subsequent speech. While this design facilitates low-latency streaming, it imposes a strict constraint where the text generation rate must exceed that of speech to ensure the transcription leads the audio. Alternatively, the Thinker-Talker architecture employs a decoupled two-stage process: a thinker model first generates text and intermediate representations, which a talker model then converts into speech tokens, functioning similarly to a TTS system. This ensures the resulting speech is semantically aligned with the text. Despite their architectural differences, both paradigms share a fundamental characteristic: the generated text is tightly coupled with the speech output, effectively serving as its script or transcription. Our work fundamentally diverges from this paradigm. We equip the SLM with intrinsic reasoning capabilities and incorporate a dedicated think-aloud module to verbalize the generated reasoning content. Instead of treating text merely as a rigid script for synthesis, our module transforms the internal Chain-of-Thought into natural, conversational speech, thereby making intermediate reasoning audible during inference.

5.2.2 Reasoning in Spoken Language Models

Integrating CoT reasoning into SLMs remains challenging, as it requires balancing reasoning capability, modality alignment, and low-latency interaction. Early efforts [Xie et al., 2025a; Wen et al., 2025; Li et al., 2025a] primarily conduct reasoning in text form, which often limits their ability to support fluid spoken dialogue or incurs substantial delay under a serial think-then-speak paradigm. To mitigate this issue, recent approaches such as STITCH and Mini-Omni-Reasoner [Chiang et al., 2025; Xie et al., 2025b] reduce latency by interleaving reasoning and spoken-response generation during inference, allowing intermediate reasoning to be incorporated while speech is produced incrementally. These methods typically realize thinking and speaking within a predefined interleaved generation schedule, such as chunk-level alternation or token-level interleaving.

Related parallel-processing paradigms have also been explored. For example, Shih et al. [Shih et al., 2025] propose a Thinking-while-Listening framework that reduces response delay by initiating text-based reasoning before the user has finished speaking. In a related but distinct direction, our framework focuses on thinking while speaking: it adopts a two-stream asynchronous design in which a reasoning thinker continuously advances the main reasoning trajectory, while a separate think-aloud module verbalizes intermediate content for a unified talker to synthesize. By explicitly decoupling reasoning progression from speech delivery, our approach enables the two stream to proceed concurrently rather than through step-by-step interleaving. Crucially, guided by a dynamic balance strategy, our system adapts to varying inference speeds without relying on fixed generation ratios. As a result, the proposed Think-Aloud mechanism externalizes intermediate reasoning as audible speech, reducing prolonged silent gaps and improving the responsiveness of spoken interaction across diverse deployment conditions.

5.3 Method

We propose a unified architecture designed to emulate the human cognitive process of "thinking while speaking." As illustrated in Figure 5.2, the architecture comprises three jointly optimized components: the reasoning thinker, the think-aloud module, and the unified talker. Complementing this architecture, we propose a dynamic balance algorithm to synchronize reasoning latency with the duration of think-aloud responses during inference.

5.3.1 The Reasoning Thinker

The reasoning thinker serves as the cognitive core of our system. It accepts the system prompt and the audio representations labeled as "User Hidden 1" in Figure 5.2 from the audio encoder as input. Similar to text-based LLMs with reasoning capabilities [Guo et al., 2025; Wei et al., 2022], the reasoning thinker firstly generates a reasoning trajectory. Subsequently, prior to generating the final response, the think-aloud responses produced by the think-aloud module are appended to the reasoning context. This ensures that the final response seamlessly aligns with the think-aloud content, facilitating a more coherent generation process.

Let $\mathbf{H}^{user}$ denote the sequence of hidden state representations corresponding to the user input, which is labeled as "User Hidden 2" in Figure 5.2. To mitigate the latency of reasoning generation, we introduce a special token `<TA_trigger>` that activates the think-aloud module. Immediately upon processing $\mathbf{H}^{user}$, the reasoning thinker generates the first `<TA_trigger>`, prompting the think-aloud module to produce an initial response before the reasoning process commences. As the thinker proceeds to generate the reasoning trajectory, it predicts additional `<TA_trigger>` tokens at specific positions, allowing the think-aloud to update its response based on the evolving reasoning context. Crucially, the reasoning thinker and the think-

aloud module operate asynchronously; the thinker continues to generate subsequent reasoning tokens in the background, running in parallel with the think-aloud module.

5.3.2 The Think-Aloud Module

The think-aloud module is a 0.5B lightweight LLM designed to generate responses that align with both user input and reasoning content. It processes context progressively, ensuring that the generated response remains coherent with the user query, previous think-aloud responses, and the ongoing reasoning process. Because the reasoning thinker and the think-aloud module operate in different semantic spaces with distinct dimensions, we introduce a projection layer, $\phi_{in} : \mathbb{R}^{d_{thinker}} \rightarrow \mathbb{R}^{d_{TA}}$.

Let k denote the index of the current think-aloud response. The input $\mathbf{X}_k^{TA}$ for the token sequence of the k-th response is constructed as follows:

- **For the 1st think-aloud response ($k = 1$)**: This response is triggered immediately after the user input. Consequently, the input consists solely of the projected representation of the user input:

$$\mathbf{X}_1^{TA} = \phi_{in}(\mathbf{H}^{user})$$

- **For subsequent think-aloud responses ($k > 1$)**: These responses are triggered upon the completion of specific reasoning segments. The input aggregates the original user input, the history of prior think-aloud utterances, and the cumulative reasoning hidden states:

$$\mathbf{X}_k^{TA} = \left[\mathbf{X}_{k-1}^{TA}; Y_{k-1}^{TA}; \phi_{in}(\mathbf{H}_{k-1}^{r})\right]$$

  where $\mathbf{H}_i^r$ (denoted as "Reasoning Hidden" in Figure 5.2) represents the reasoning segment generated between the i-th and $(i+1)$-th `<TA_trigger>` token,

and $\mathbf{Y}_i^{TA}$ (labeled as "TA Response Token" in Figure 5.2) denotes the token sequence of the i-th think-aloud response.

Before the talker generates the final response, the think-aloud response generated by the think-aloud module is appended to the reasoning content, ensuring that the final response generated by the thinker is consistent with the user's question, the reasoning content, and the think-aloud responses.

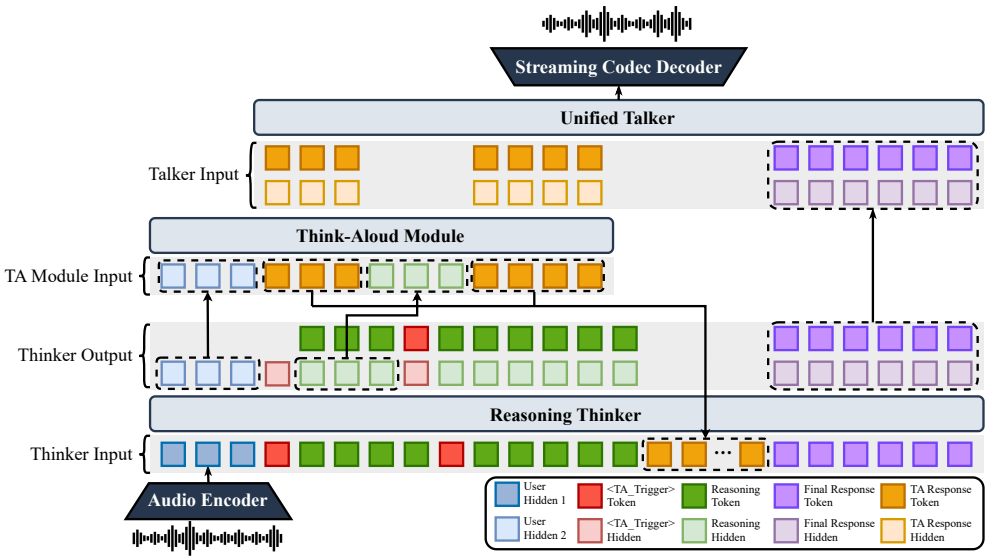


Figure 5.2: The overall architecture of our proposed model, illustrating the collaborative workflow between the reasoning thinker, the think-aloud module, and the unified talker. "→" denotes the copy operation. The dashed box indicates the source and target of the copy mechanism.

5.3.3 The Unified Talker

To synthesize continuous speech from the outputs of both the reasoning thinker and the think-aloud module, we employ CosyVoice 2.0 [Du et al., 2024] in streaming mode as a unified talker. To integrate linguistic information (text embeddings) with high-level guidance (hidden states), we introduce two distinct linear projection heads that

map the hidden states into the talker's input dimension (d_{talk}). Specifically, $\phi_{out}^{th}(\cdot)$ maps $\mathbb{R}^{d_{thinker}} \to \mathbb{R}^{d_{talk}}$, and $\phi_{out}^{TA}(\cdot)$ maps $\mathbb{R}^{d_{TA}} \to \mathbb{R}^{d_{talk}}$.

At each generation step, the input $\mathbf{u}$ to the unified talker is constructed by summing the native CosyVoice text embedding $\mathbf{e}_{text}$ and the aligned representation $\mathbf{r}_r$. The source of this representation switches dynamically based on the current generation phase:

$$\mathbf{u} = \mathbf{e}_{text} + \mathbf{r}_r$$

where $\mathbf{r}_r$ is defined as:

$$\mathbf{r}_r = \begin{cases} \phi_{out}^{TA}(\mathbf{H}^{TA}) & \text{if in reasoning phase} \\ \phi_{out}^{th}(\mathbf{H}^{th}) & \text{if in final response phase} \end{cases}$$

During the reasoning phase, the talker synthesizes the think-aloud response driven by the think-aloud module's hidden states $\mathbf{H}^{TA}$ (labeled as "TA Response Hidden" in Figure 5.2) to bridge the silence. Once reasoning is complete, the model transitions to the final response phase, where the talker utilizes the thinker's hidden states $\mathbf{H}^{th}$ (labeled as "Final Response Hidden" in Figure 5.2) to deliver the final answer.

To ensure robust alignment and compatibility with CosyVoice 2.0, we adhere to the streaming data formatting protocols of CosyVoice 2.0. The inputs and outputs for the talker are prepared as interleaved sequences of input representations and output tokens at a ratio of 5 to 15.

5.3.4 Training Objective

To jointly optimize the reasoning thinker, the think-aloud module, and the unified talker, we employ a multi-task training objective composed of three terms, corresponding to the thinker generation, think-aloud generation, and talker generation,

respectively:

$$\mathcal{L} = \mathcal{L}_{Thinker} + \mathcal{L}_{TA} + \mathcal{L}_{Talk}$$

Note that the audio encoder is fixed during training.

5.3.5 Dynamic Balance Strategy

During training, the model learns to trigger appropriate think-aloud responses based on reasoning milestones, as described in Section 5.4. However, during real-time streaming inference, the physical duration of synthesized speech may not align with the varying computation time required for reasoning generation. To reduce long silent gaps and excessive articulation overhead, we introduce a state-driven dynamic balance strategy to coordinate the asynchronous reasoning (Thinker) and speaking (think-aloud module) streams.

Rather than relying on pre-computed time estimates, the proposed strategy tracks the runtime states of both streams and is invoked at key transition events, e.g., when the currently active think-aloud playback finishes while reasoning is still ongoing, or when the reasoning stream reaches completion. It handles two primary boundary scenarios:

- **Audio starvation (reasoning is ongoing, but speech playback has finished):** This scenario occurs when the reasoning process is still generating tokens, but no think-aloud audio remains available for playback. To reduce user-perceived silence, the strategy scans the current reasoning buffer for the most recently completed logical segment that has not yet triggered a response, using the same double-newline-based segment boundary convention as in training. It then issues a new think-aloud trigger for that segment, providing additional audible feedback while reasoning continues.

- **Reasoning early completion (reasoning finishes while think-aloud is**

ongoing): This scenario arises when the thinker completes its deduction before all potential think-aloud responses have been synthesized. To produce the final response, the strategy immediately cancels all *pending* (i.e., not yet synthesized) `<TA_trigger>` tokens. **The *currently active* think-aloud utterance, however, is allowed to finish before the system proceeds to final-response generation.** The completed utterance is appended to the reasoning context so that the subsequent final response remains consistent with the speech already presented to the user.

The empirical effect of this strategy is analyzed in Section 5.6.2. In particular, the latency results in Figure 5.3 show that the resulting unmasked silence remains very low for most test samples. In addition, Case 1 in Section 5.6.4 further illustrates this strategy.

5.4 Data Preparation

In this section, we describe the methodology for preparing our reasoning and "think-aloud" datasets using in-house SFT data, which contains both single-turn and multi-turn spoken dialogue data.

5.4.1 Reasoning Data Generation

Our reasoning data is generated via a three-step pipeline, which utilizes the Deepseek-R1 model [Guo et al., 2025] for all generation and filtering tasks.

- We first identify and select reasoning-intensive turns from the in-house SFT data. We evaluate each turn five times using the identical prompt, and the final result is determined by a majority vote.

- Second, we prompt Deepseek-R1 with the dialogue history and the current user input for each selected turn. We then extract the resulting reasoning text while discarding the model's final generated reply.

- Finally, we validate the generated reasoning text. Since the extracted reasoning text is conditioned only on the dialogue history and user input, its alignment with the final ground-truth answer is not guaranteed. Therefore, we apply an additional filtering pass to ensure this consistency and filter out any mismatches. Similar to the first step, we run five times using the same prompt and decide the answer by a majority vote.

5.4.2 Think-Aloud Response Generation

For the generation of think-aloud responses, we employ a process with four steps. All steps use the Deepseek-R1 model [Guo et al., 2025].

- First, the reasoning text is segmented into multiple paragraphs using double newline characters as delimiters. Following segmentation, we identify potential trigger points for the think-aloud responses. Our criterion here is whether a preliminary conclusion can be drawn from the reasoning content of the current segment. To ensure the stability of this process and filter out spurious triggers, the identification routine is executed five times, and the final set of triggers is determined via a majority vote.

- Once the triggers are finalized, the model generates the think-aloud responses. This generation is conditioned on both the original user input and the specific reasoning content associated with the validated trigger points. We intentionally restrict each think-aloud response to a single sentence to keep the verbalized reasoning concise, controllable in duration, and less likely to dominate the final answer.

- In the third step, to ensure consistency between the final response and the think-aloud responses within the original SFT dataset, we revise the final response accordingly.
- Finally, we utilize the internal TTS to resynthesize speech for all responses, ensuring acoustic consistency. This process covers the entire dialogue history, the think-aloud response, and the final responses. All generated speech segments are unified into a single-speaker voice.

5.5 Experimental Setup

5.5.1 Implementation Details

5.5.1.1 Data

We utilize an in-house dataset comprising approximately 200,000 single-turn and multi-turn dialogues, totaling around 5,000 hours of speech. To provide context on the data distribution, these dialogues span a diverse range of scenarios, including general commonsense QA, reasoning and logical deduction, as well as general helpfulness queries. This composition ensures the model's robust generalization across various spoken interactions. The reasoning text and think-aloud responses are prepared following the methodology outlined in Section 5.4.

5.5.1.2 Model Training

For model initialization, we leverage several pre-trained models. The audio encoder and reasoning thinker are initialized using the parameters from the Qwen2.5-Omni-7B [Xu et al., 2025] thinker model and audio encoder. The think-aloud module is initialized with Qwen2.5-0.5B-Instruct [Yang et al., 2024c]. The unified talker and

streaming codec decoder, which are responsible for generating the final audio, are initialized using the CosyVoice 2.0 LLM and the flow matching model [Du et al., 2024].

The entire model is trained for 10,000 steps on 64 H100 GPUs using a batch size of 64. We employ a learning rate scheduler with a peak value of 1×10^{-5}, which includes a 500-step warm-up phase followed by an exponential decay. During model training, we randomly select a turn containing reasoning data from the dialogue to designate as the final turn. Concurrently, we randomly select either one of the think-aloud responses or the final system response to train the talker.

5.5.2 Evaluation

We evaluate our models on two categories of datasets. First, to assess reasoning capabilities, we utilize the single-step and multi-step reasoning subsets of the Spoken-MQA benchmark [Wei et al., 2025]. Correctness is assessed by a GPT-4o judge using a best-of-three majority voting strategy, as standard rule-based metrics proved unreliable due to normalization errors. Second, to test commonsense knowledge and factuality, we use the Web Questions [Berant et al., 2013], and TriviaQA [Joshi et al., 2017]. For this category, we apply the exact match to determine if the ground-truth answer was present in the model's response. The metric is accuracy for both the Web Question and TriviaQA test set. In our evaluation, the Llama questions [Nachmani et al., 2023] dataset is also considered. We find that the performance difference between reasoning-enabled and non-reasoning models is negligible on this specific benchmark. Given this lack of differentiation, the dataset is ultimately omitted from our final suite of experiments.

5.6 Experimental Results

5.6.1 Main Results

To evaluate the effectiveness of our approach, we benchmark against two control settings. The baseline represents a model without the think-aloud module, fine-tuned solely on direct-response data, i.e., data that excludes intermediate reasoning steps. The baseline with reasoning uses the same backbone but is trained on datasets augmented with reasoning text. This model follows a serial paradigm, generating the full textual reasoning before synthesizing speech, and thus serves as a strong serial reasoning reference, albeit with substantially higher latency.

The two controls serve different purposes. The baseline represents a non-reasoning spoken assistant and is used to measure the gap between our full reasoning-enabled system and a direct-response spoken model. The baseline with reasoning is the more relevant design-level control, since it shares reasoning-augmented supervision with our method but follows a serial think-then-speak pipeline.

Table 5.1: Performance on single-step and multi-step reasoning subsets of Spoken-MQA benchmark, measured in accuracy.

Models	**Single-Step**	**Multi-Step**	**Average**
Whisper-Qwen2.5-7B-Instruct	81.0	68.9	75.0
Whisper-Deepseek-Math-7B-instruct	85.2	78.2	81.7
Whisper-Qwen2.5-Math-7B-Instruct	88.0	86.2	87.1
LLaMA-Omni-7B [Fang et al., 2024]	29.5	10.5	20.0
Qwen2-Audio-7B-Instruct [Chu et al., 2024]	56.2	19.2	37.7
Freeze-Omni [Wang et al., 2024d]	69.0	19.8	44.4
GLM-4-Voice [Zeng et al., 2024]	54.4	28.5	41.5
Mini-Omni-Reasoner [Xie et al., 2025b]	85.9	60.5	68.1
Baseline	86.0	63.0	74.5
Baseline with Reasoning	91.3	85.6	88.5
Our Proposed Model	91.6	83.6	87.6

Table 5.1 presents the evaluation results on the Spoken-MQA benchmark, which assesses the model's ability to handle both single-hop and multi-hop reasoning in speech. Compared with the direct-response baseline, our full system achieves substantially higher accuracy on reasoning-intensive tasks, especially on the multi-step subset. This comparison reflects the performance gap between a non-reasoning spoken assistant and our full reasoning-enabled system. More importantly, when compared with the baseline with reasoning, our method achieves highly comparable performance (87.6% vs. 88.5% on average). This suggests that our framework largely preserves the logical depth and answer accuracy of CoT reasoning, while avoiding the long silent delays of the serial paradigm.

Table 5.2: Speech-to-Speech performance on spoken QA benchmarks, measured in accuracy.

Models	**Web Questions**	**TriviaQA**	**Average**
GPT-4o-Realtime [Hurst et al., 2024]	51.6	69.7	60.7
Moshi [Défossez et al., 2024]	9.2	7.3	8.3
Mini-Omni [Xie and Wu, 2024]	12.8	6.9	9.9
GLM-4-Voice [Zeng et al., 2024]	15.9	26.5	21.2
LUCY (S2) [Gao et al., 2025]	25.6	22.9	24.3
Freeze-Omni [Wang et al., 2024d]	26.1	25.7	25.9
LLaMA-Omni2-7B [Fang et al., 2025]	31.3	-	-
MinMo [Chen et al., 2025]	39.9	37.5	38.7
MiniCPM-o 2.6 [Yao et al., 2024]	40.0	40.2	40.1
VITA-Audio [Long et al., 2025]	41.7	42.7	42.2
Baseline	32.1	36.1	34.1
Baseline with Reasoning	40.3	39.2	39.8
Our Proposed Model	40.3	38.7	39.5

We further evaluate general knowledge performance on the Web Questions and TriviaQA datasets under both Speech-to-Speech (Table 5.2) and Speech-to-Text (Table 5.3) settings. Consistent with the findings on Spoken-MQA, our model outperforms the direct-response baseline by a clear margin (e.g., +5.4% average accuracy in

Table 5.3: Speech-to-Text performance on spoken QA benchmarks, measured in accuracy.

Models	**Web Questions**	**TriviaQA**	**Average**
Moshi [Défossez et al., 2024]	26.6	22.8	24.7
LUCY (S2) [Gao et al., 2025]	29.3	27.0	28.2
GLM-4-Voice [Zeng et al., 2024]	32.2	39.1	35.7
LLaMA-Omni2-7B [Fang et al., 2025]	34.5	-	-
VITA-Audio [Long et al., 2025]	45.0	45.9	45.5
Baseline	33.7	39.6	36.7
Baseline with Reasoning	42.5	42.1	42.3
Our Proposed Model	43.0	41.5	42.3

S2S) and remains highly comparable to the baseline with reasoning. Taken together, these results suggest that concurrent think-aloud generation does not materially degrade the factual accuracy of the final response.

Tables 5.1, 5.2, and 5.3 also include results from recent state-of-the-art SLMs for reference. Although direct comparison is difficult due to differences in training data and experimental settings, the results indicate that our method remains competitive relative to contemporary spoken language models.

5.6.2 Latency Analyses

In real-world deployments, SLMs operate under diverse hardware configurations and software environments, leading to significant fluctuations in inference speed. Consequently, a robust think-aloud mechanism must adaptively synchronize the duration of generated speech with the varying time costs of reasoning. To evaluate this temporal synchronization, we introduce two specific latency metrics: Unmasked Silence Latency (L_{sil}), which measures the cumulative duration of "dead air" where the user waits without audio feedback during reasoning. And Articulation Overhead Latency (L_{oh}), which captures the delay caused by excessive think-aloud responses continuing

after the reasoning process has concluded.

Table 5.4: Performance and latency analysis on the Spoken-MQA benchmark. **Accuracy** is reported on single-step and multi-step reasoning subsets. **Latency** includes unmasked silence latency L_{sil} and articulation overhead latency L_{oh}. "DB Stgy." is short for dynamic balance strategy.

Model / Speed (tok/s)	**Accuracy (%)** ↑		**Latency (s)** ↓	
	Single	**Multi**	L_{sil}	L_{oh}
Baseline	86.0	63.0	-	-
Baseline with Reasoning	91.3	85.6	12.82	-
Our Model w/o DB Stgy.	90.7	84.5	1.03	19.92
Our Proposed Model (varying generation speed)				
40	91.6	83.6	0.36	4.34
80	91.6	84.2	0.05	4.16
160	91.8	84.3	0.03	5.66

As presented in Table 5.4, the experimental results demonstrate the critical role of our proposed dynamic balance strategy in optimizing these metrics. The baseline with reasoning achieves competitive accuracy (85.6% on Multi-step tasks) but suffers from a prohibitive L_{sil} of 12.82s, confirming that serial "think-then-speak" pipelines are impractical for fluid interaction. Conversely, the ablation variant (Our Model w/o DB Stgy.) reveals the risks of unregulated generation: while it successfully mitigates silence ($L_{sil} \approx 1.0s$), it fails to align speech duration with reasoning cost, resulting in a severe articulation overhead (L_{oh}) of 19.92s. This indicates that without dynamic control, the system tends to generate redundant fillers that unnecessarily prolong the turn.

In contrast, our proposed model achieves an optimal balance across varying generation speeds (40, 80, and 160 tokens per second). By dynamically adjusting the speech budget, our method significantly suppresses both latency types while maintaining accuracy comparable to the reasoning baseline. Notably, at a standard speed of 80 tok/s, we achieve a negligible L_{sil} of 0.05s and reduce L_{oh} to 4.16s, which is a

substantial improvement over the uncontrolled variant.

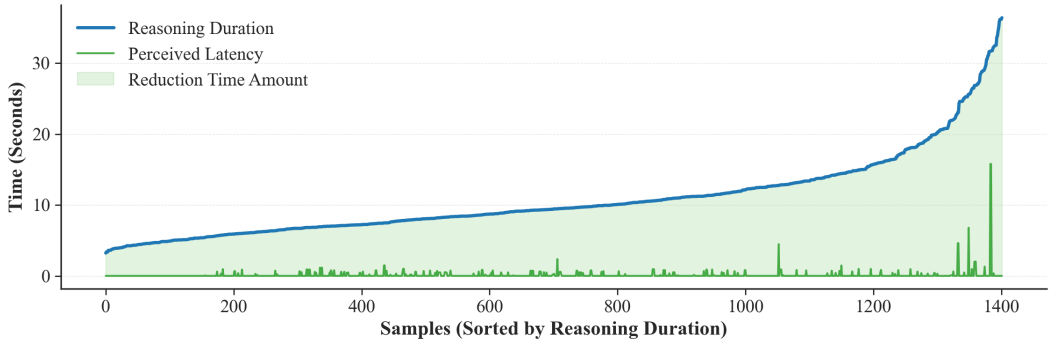


Figure 5.3: Impact of think-aloud utterances on perceived latency in spoken dialogue. The plot illustrates the Reasoning Duration (blue solid line) and the corresponding Perceived Latency experienced by the user (green line) across test samples, sorted in ascending order of Reasoning Duration. The light green shaded region, denoted as the Reduction Time Amount, highlights the duration of system silence effectively eliminated by the model's think-aloud mechanism. The results demonstrate that generating think-aloud utterances successfully masks the underlying processing time, maintaining a near-zero perceived latency even for samples requiring extensive reasoning.

To further analyze the objective latency behavior of the proposed mechanism, we visualize the temporal dynamics of the system using the multi-step subset of the Spoken-MQA benchmark. As illustrated in Figure 5.3, we compare the absolute model Reasoning Duration (blue line) against the Perceived Latency experienced by the user (green line). For clarity, the test samples are sorted in ascending order based on their reasoning duration.

As the thinker module actively performs reasoning, the think-aloud module continuously generates think-aloud utterances, effectively masking the underlying processing time. As shown by the shaded region (Reduction Time Amount), the system successfully eliminates a substantial portion of the silence. Consequently, even for complex queries that require extensive reasoning (exceeding 30 seconds), the perceived latency generally remains near zero. While a negligible fraction of outlier cases still exhibit a perceived latency of around 10 seconds, the vast majority of in-

teractions successfully maintain a near-zero wait time, ensuring a highly responsive system with continuous acoustic feedback overall.

Table 5.5: Performance in terms of speech quality of Qwen2.5-Omni and our proposed model on the single-step reasoning subset of Spoken-MQA benchmark.

Models	**STOI** ↑	**PESQ** ↑	**SI-SDR** ↑	**NISQA** ↑
Qwen2.5-Omni	0.99	3.69	24.71	4.65
Our Proposed Model	0.99	3.97	24.50	4.95

5.6.3 Speech Quality

To evaluate speech quality, we compare our proposed model against Qwen2.5-Omni [Xu et al., 2025] on the single-step reasoning subset of the Spoken-MQA benchmark [Wei et al., 2025]. We utilize four metrics: STOI, PESQ, SI-SDR, and NISQA. These metrics are calculated using TorchAudio-Squim [Kumar et al., 2023], and as shown in Table 5.5, higher scores indicate better performance.

Overall, the proposed model achieves intelligibility comparable to that of Qwen2.5-Omni. However, it provides notably better perceptual quality, improving PESQ from 3.69 to 3.97 and NISQA from 4.65 to 4.95. Conversely, the SI-SDR score is slightly lower for the proposed model. Taken together, these results demonstrate that the speech quality of our proposed model is comparable to that of Qwen2.5-Omni.

5.6.4 Qualitative Examples

To qualitatively illustrate the behavior of our proposed model, we present two representative interaction examples.

Case 1 demonstrates the model's standard operating mode, as shown in Figure 5.4. During the thinker reasoning process, a total of three `<TA_trigger>`s are generated. However, because the first one is sufficient to cover the reasoning time cost, the sec-

ond and third triggers are canceled. Therefore, instead of remaining silent for a long period and then producing a direct answer, the system first generates a think-aloud response (highlighted in light blue). By verbalizing intermediate arithmetic steps, the model reduces the silent gap between the query and the final solution, providing the user with earlier audible feedback before delivering the final response (highlighted in dark blue).

Case 1 (Explicit Reasoning)
User Input:
Paige raised 7 goldfish and 12 catfish in the pond but stray cats loved eating them. Now she has 15 left. How many fishes disappeared?
System Reasoning:
<TA_trigger>Okay, let's see. Paige started with 7 goldfish and 12 catfish. So the total number of fish she had initially would be 7 plus 12. Let me add those: 7 + 12 is 19. So she had 19 fish in total.

Now, after some time, she has 15 fish left. The question is asking how many fish disappeared. To find out, I need to subtract the number of fish left from the original number. So that would be 19 minus 15. Let me do that calculation: 19 - 15 equals 4.

~~<TA_trigger>~~Wait, but let me double-check. If she started with 19 fish and now has 15, then yes, 19 - 15 is 4. That means 4 fish disappeared. But wait, the problem mentions that stray cats loved eating them. So the fish that disappeared would be eaten by the cats. So the answer should be 4 fish disappeared.

~~<TA_trigger>~~ Hmm, but let me make sure there's no trick here. The problem doesn't mention any other factors, like fish dying naturally or anything else. It just says the stray cats loved eating them. So the disappearance is due to the cats. So the answer is 4. Yeah, that makes sense.
System Response:
Okay, I need to figure out how many fish disappeared from Paige's pond after she started with 7 goldfish and 12 catfish, ending up with 15 left. So, Paige started with 19 fish and now has 15 left, meaning 4 fish disappeared.

Figure 5.4: Qualitative case study demonstrating thinking while speaking. This case shows the model verbalizing its thought process to bridge the gap between the query and the solution. Gray backgrounds represent User Input. Light Blue denotes the think-aloud responses, while Dark Blue highlights the final response derived from the reasoning process. The `<TA_trigger>` with a strike-through indicates that the trigger is canceled by the dynamic balance strategy.

Case 2 presents a simulated interruption scenario for error correction, as shown in Figure 5.5. Case 2 is included as an illustrative future-use scenario rather than as a validated capability of the current system. During a complex reasoning task, the model initially misinterprets a unit constraint, confusing time (“64 seconds”) with distance (“64 feet”). In a conventional text-based system, the model typically thinks and then provides a response, but because the thinking process is lengthy, it is often

not very readable. Users can also interrupt when they discover a problem, but this is usually done by editing the original question. By contrast, because our framework externalizes intermediate reasoning as speech, such an error could potentially be noticed earlier and more easily by the user. In this illustrative example, we assume an external interruption mechanism (e.g., VAD or turn-taking control), which is outside the scope of this work and is not implemented or quantitatively evaluated here. Under this assumption, the user interrupts after detecting the mistake, and the second turn shows how the interaction could proceed after the correction is provided.

Case 2 (Barge-in Error Correction)
User Input:
Julia's boat sprang a leak while she was out on the lake. The boat was taking on two liters of water for every ten feet she rowed back towards shore. It took her sixteen seconds to row twenty feet. The shore was 64 seconds away. How much water had the boat taken on by the time Julia reached shore?
System Reasoning:
<TA_trigger>Okay, let's see. Julia's boat leaked, and I need to figure out how much water it took on by the time she reached the shore. Hmm. Let's break down the problem step by step.

First, the boat is taking on 2 liters of water for every 10 feet she rows back towards shore. So, the rate of water intake is 2 liters per 10 feet. That means for every 10 feet she rows, 2 liters are added to the boat.

Then, it took her 16 seconds to row 20 feet. Wait, so her rowing speed is 20 feet in 16 seconds. But the shore is 64 seconds away. Wait, that might be a bit confusing. Let me check the units. If she rows 20 feet in 16 seconds, then her speed is 20/16 feet per second. But the distance to the shore is 64 seconds away? Wait, that doesn't make sense. Wait, maybe the shore is 64 feet away, not seconds. Because 64 seconds would be time, not distance. So the shore is 64 feet away. Let me confirm that. The problem says "the shore was 64 seconds away." That's probably a typo. It should be 64 feet. So she needs to row 64 feet to reach the shore.

<TA_trigger>
System Response:
Okay, I need to calculate how much water Julia's boat took in while rowing back to the shore after she got stuck. First, I'll define the variables: the boat's speed, the leak rate, the distance to the shore, and the time to row. I've realized the distance to the shore is actually 64 feet, not 64 seconds.
User Input:
Small correction, the “64” given is in seconds, indicating how long it takes to reach shore, not how many feet remain.
System Response:
Got it, you‘re clarifying that the 64 seconds given is the time to reach shore, not the distance remaining. Okay, so the leak rate is 2 liters per 10 feet, and I need to figure out the total distance covered. First, I’ll calculate her rowing speed. She rowed 20 feet in 16 seconds, which gives 1.25 feet per second. Since the time to shore is 64 seconds, the total distance to shore is 64 seconds multiplied by 1.25 feet per second, which is 80 feet. Now, looking at the leak rate of 2 liters per 10 feet, 80 feet contains eight of those 10-foot segments, so eight times 2 liters equals 16 liters. So putting that together, Julia's boat took on 16 liters of water by the time she reached.

Figure 5.5: Case 2 illustrates a simulated interruption scenario for error correction. *This example is intended to show a possible deployment setting in which exposed think-aloud speech may allow earlier user intervention when an error is detected in the reasoning stream.* The scenario assumes an external interruption mechanism (e.g., VAD or turn-taking control), which is not implemented or evaluated in this work.

5.6.5 Human Evaluation

To complement the objective evaluations, we conduct a small-scale human evaluation on 50 randomly sampled test cases from the multi-step subset of Spoken-MQA benchmark, with 3 annotators per case. For each sample, annotators compare the output from our proposed model with the output that remains silent during thinking and only outputs the final response. In a blind A/B setting, they are asked to choose (1) which system they prefer overall and (2) which system feels more responsive. In addition, they rate the fluency/naturalness of the proposed model output on a 5-point Likert scale (1: very unnatural, 3: Acceptable 5: very natural).

The proposed model is preferred overall in 85.3% of the cases and is judged to be more responsive in 99.3% of the cases, while achieving a fluency/naturalness score of 4.01. Qualitatively, we observe that the inserted think-aloud responses are generally consistent with the final responses, and the transition to the final answer is typically smooth rather than abruptly cut off. These results provide preliminary evidence that the think-aloud mechanism improves perceived responsiveness while maintaining good naturalness.

5.7 Summary

This chapter presented a think-aloud mechanism to address the tension between complex reasoning latency and the responsiveness required in spoken interaction. By introducing a Thinker-Talker architecture with a novel think-aloud module that coordinates asynchronous reasoning and speaking streams, our framework enables spoken language models to externalize part of their intermediate reasoning as speech during inference. Experiments demonstrate that our approach substantially reduces long silent latency and improves turn responsiveness, while preserving most of the reasoning benefits of a serial think-then-speak system. Overall, our method provides a

practical design framework for building more responsive spoken reasoning agents. As a direction for future work, we note that the current system does not explicitly constrain the length of internal reasoning; future research may explore reinforcement learning or related optimization methods to adaptively shorten reasoning, or skip it altogether when unnecessary, in order to further improve inference efficiency.

□ End of chapter.

Chapter 6

Conclusion and Future Work

Summary

This chapter summarizes the contributions of this dissertation and proposes potential directions for future work.

6.1 Contributions

This dissertation has studied speech intelligence across multiple levels of abstraction, under the theme of *representation learning and cognitive modeling of speech in real-world scenarios*. Rather than focusing on a single end-to-end architecture, it has investigated a connected research agenda that progresses from foundational speech representation learning, to robust modeling under realistic acoustic conditions, and finally to cognitively capable spoken language systems. In this sense, the dissertation follows a coherent progression from speech representation learning, to robust speech modeling, to beyond-word perception, and ultimately to spoken reasoning and interaction.

The first contribution of this dissertation lies in speech-only self-supervised pre-training with discrete units. Chapter 2 showed that discrete speech codes provide an effective bridge between raw continuous speech and higher-level symbolic pre-training objectives when paired text supervision is unavailable. On the encoder side, CoBERT demonstrated that speech representations can be improved by predicting contextualized code representations from masked input. On the decoder side, Speech2C further extended speech-only pre-training to encoder–decoder ASR models by using pseudo codes as reconstruction targets. Together, these studies provided a unified perspective on how discrete units can support both encoder representation learning and decoder pre-training under speech-only conditions.

The second contribution concerns robust speech modeling in real-world multi-speaker scenarios. Chapter 3 argued that effective speech intelligence requires not only informative representations, but also robustness to overlap, interference, and speaker ambiguity. To this end, the dissertation studied speaker-aware self-supervised learning for mixture speech and presented a unified framework for speaker extraction and diarization. These efforts extended speech modeling from clean and isolated settings toward more realistic acoustic environments, and established an important bridge between foundational speech representation learning and practical real-world speech systems.

The third contribution of this dissertation is the study of perception beyond words in spoken language models. Chapter 4 moved beyond transcript-based understanding and investigated how spoken systems can perceive richer information carried in speech, including paralinguistic cues and acoustic context. In particular, the dissertation introduced SD-Eval as a benchmark for spoken dialogue understanding beyond lexical content, and presented SOLLA as a speech-oriented large language model that incorporates acoustic context into spoken interaction. These studies highlighted that speech should not be treated merely as an acoustic signal for transcription, but as

a rich modality containing information that is crucial for more natural and context-aware spoken interaction.

The final contribution lies in low-latency spoken reasoning. Chapter 5 addressed the tension between complex reasoning and responsiveness in spoken interaction. By introducing a Thinker-Talker architecture with a think-aloud module, the dissertation proposed a framework in which reasoning and speaking proceed asynchronously, thereby reducing prolonged silence while preserving strong reasoning ability. This work further extended the dissertation from spoken perception to spoken reasoning, and provided a practical design perspective for building more responsive spoken language systems.

Taken together, the studies in this dissertation present a unified view of speech intelligence across multiple levels: learning effective representations from speech-only data, building robust speech systems for realistic acoustic conditions, enabling spoken language models to perceive information beyond words, and supporting low-latency spoken reasoning and interaction. It is hoped that these efforts can contribute to the development of more natural, robust, and intelligent speech-based human–computer interaction systems.

6.2 Future Work

Although this dissertation has explored several important aspects of speech intelligence, many open problems remain. A first direction for future work is to further unify the different levels studied in this dissertation. The earlier chapters focused on representation learning and robustness, whereas the later chapters focused on perception and reasoning in spoken language models. A promising next step is therefore to build more unified speech-language systems that integrate low-level acoustic modeling, robust speech understanding, contextual perception, and high-level reasoning

within a single framework. Such a direction may help reduce the gap between speech encoders and speech-oriented large language models, and enable more coherent end-to-end spoken intelligence.

A second direction is to move toward more realistic and richer spoken interaction settings. In Chapter 4, the benchmark and modeling studies already emphasized the importance of information beyond lexical content. However, future work could further extend this line in several ways, including multi-turn speech-to-speech dialogue, richer paralinguistic dimensions, and more comprehensive contextual understanding. For example, future benchmarks may consider additional speaker attributes, more dynamic conversational settings, and tighter evaluation of response appropriateness in realistic spoken interaction. These extensions would be important for building spoken systems that are not only accurate, but also adaptive, empathetic, and context-aware.

A third direction is to improve real-time spoken reasoning. As discussed in Chapter 5, the current think-aloud framework improves responsiveness by exposing part of the reasoning process as speech, but it does not explicitly constrain the length of internal reasoning. Future work may explore reinforcement learning or related optimization strategies to adaptively shorten reasoning, control when think-aloud responses should be generated, or even skip explicit reasoning when it is unnecessary. More generally, a practical spoken reasoning system should learn to balance reasoning quality, interaction latency, and speech naturalness under different user demands and deployment conditions.

Finally, a broader future direction is to extend the scope of this dissertation toward multimodal and real-world interactive intelligence. Human communication rarely depends on speech alone; it often involves visual context, environmental signals, user state, and interaction history. Future work may therefore investigate multimodal spoken systems that jointly model speech, text, audio events, visual information, and dialogue memory. In addition, on-device deployment, cross-lingual generalization, and

personalized spoken interaction are all promising directions that could make speech intelligence more practical and widely accessible in real-world applications.

In summary, this dissertation has taken several steps toward understanding and building speech intelligence from representation learning to cognitive spoken interaction. Future research will continue to move in the direction of more unified, robust, context-aware, and interactive spoken language systems.

□ End of chapter.

Appendix A

Complete Publication List During Ph.D. Study

The following is a complete list of publications authored, accepted, or currently under review during my Ph.D. study. Unlike the chapter-wise list in Chapter 1, this appendix includes both dissertation-related work and other research outputs from the same period.

1. **Ao, J.**, Peng, K., Liu, X., Zhang, S., Tang, Z., Ma, X., Li, X., Wang, Y., Wu, Z., Li, H., He, Q., and Ma, M. (2026b). Spoken Language Models that Think Aloud. Under review for *Interspeech 2026*.

2. **Ao, J.**, Chen, D., Tian, X., Feng, W., Zhang, J., Lu, L., Wang, Y., Li, H., and Wu, Z. (2026a). Solla: Towards a Speech-Oriented LLM That Hears Acoustic Context. Under review for *Interspeech 2026*.

3. **Ao, J.**, Yıldırım, M. S., Tao, R., Ge, M., Wang, S., Qian, Y., and Li, H. (2025). USED: Universal Speaker Extraction and Diarization. *IEEE Transactions on Audio, Speech and Language Processing*, 33:96–110.

4. **Ao, J.**, Wang, Y., Tian, X., Chen, D., Zhang, J., Lu, L., Wang, Y., Li, H., and Wu, Z. (2024). SD-Eval: A Benchmark Dataset for Spoken Dialogue Understanding Beyond Words. In *The Thirty-eighth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

5. **Ao, J.**, Zhang, Z., Zhou, L., Liu, S., Li, H., Ko, T., Dai, L., Li, J., Qian, Y., and Wei, F. (2022b). Pre-Training Transformer Decoder for End-to-End ASR Model with Unpaired Speech Data. In *Interspeech 2022*, pages 2658–2662.

6. Meng, C.*, **Ao, J***., Ko, T., Wang, M., and Li, H. (2023). CoBERT: Self-Supervised Speech Representation Learning Through Code Representation Learning. In *Interspeech 2023*, pages 2978–2982. * Equal contribution.

7. Lin, J., **Ao, J.**, Ge, M., Feng, M., and Li, H. (2026). A Two-Stage Self-Supervised Speech Representation Learning for Acoustic, Phonetic and Semantic Modeling. *IEEE Transactions on Audio, Speech and Language Processing*, pages 1–13.

8. Wang, L., **Ao, J.**, Gan, L., Wang, Y., Zhang, X., and Wu, Z. (2025). Audio Deepfake Verification. *arXiv preprint arXiv:2509.08476*.

9. Yue, X., **Ao, J.**, Gao, X., and Li, H. (2023b). Token2vec: A Joint Self-Supervised Pre-Training Framework Using Unpaired Speech and Text. In *ICASSP 2023*, pages 1–5.

10. Zhang, Z. and **Ao, J**. (2022). The YiTrans speech translation system for the IWSLT 2022 offline shared task. In *Proceedings of the 19th International Conference on Spoken Language Translation (IWSLT 2022)*, pages 158–168. Association for Computational Linguistics.

11. Lin, J., Ge, M., **Ao, J.**, Deng, L., and Li, H. (2024b). SA-WavLM: Speaker-Aware Self-Supervised Pre-Training for Mixture Speech. In *Interspeech 2024*, pages 597–601.

12. Ma, D., Yue, X., **Ao, J**., Gao, X., and Li, H. (2024b). Text-Guided HuBERT: Self-Supervised Speech Pre-Training via Generative Adversarial Networks. *IEEE Signal Processing Letters*, 31:2055–2059.

13. Lin, J., Yue, X., **Ao, J**., and Li, H. (2023). Self-Supervised Acoustic Word Embedding Learning via Correspondence Transformer Encoder. In *Interspeech 2023*, pages 2988–2992.

14. Zhang, Z., Zhou, L., **Ao, J**., Liu, S., Dai, L., Li, J., and Wei, F. (2022d). SpeechUT: Bridging Speech and Text with Hidden-Unit for Encoder-Decoder Based Speech-Text Pre-Training. In *Proceedings of EMNLP 2022*, pages 1663–1676. Association for Computational Linguistics.

15. Wang, Y., Tang, Z., Wang, Y., Hinsvark, A., Liu, Y., Li, Y. A., Peng, K., **Ao, J**., Ma, M., Seltzer, M., He, Q., and Liu, X. (2026). Scaling Speech Tokenizers with Diffusion Autoencoders. In *The Fourteenth International Conference on Learning Representations*.

16. Zhou, L., Yu, L., Lyu, Y., Lin, Y., Zhao, Z., **Ao, J**., Zhang, Y., Wang, B., and Li, H. (2026). Echomind: An Interrelated Multi-Level Benchmark for Evaluating Empathetic Speech Language Models. In *The Fourteenth International Conference on Learning Representations*.

17. Yıldırım, M. S., Tao, R., Wang, W., **Ao, J**., and Li, H. (2025). Leveraging Language Information for Target Language Extraction. In *2025 Asia Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 837–842. IEEE.

18. Li, J., Zhang, X., Wang, Y., He, H., Wang, C., Wang, L., Liao, H., **Ao, J**., Xie, Z., Huang, Y., et al. (2025b). Overview of the Amphion Toolkit (v0.2). *arXiv preprint arXiv:2501.15442*.

19. Ge, M., Peng, Y., Jiang, Y., Lin, J., **Ao, J.**, Yıldırım, M. S., Wang, S., Li, H., and Feng, M. (2023). The NUS-HLT System for the ICASSP 2024 ICMC-ASR Grand Challenge. *arXiv preprint arXiv:2312.16002*.

□ End of chapter.

Bibliography

Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. (2023). Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Adigwe, A., Tits, N., Haddad, K. E., Ostadabbas, S., and Dutoit, T. (2018). The emotional voices database: Towards controlling the emotion dimension in voice generation systems. *arXiv preprint arXiv:1806.09514*.

Anderson, P., Fernando, B., Johnson, M., and Gould, S. (2016). Spice: Semantic propositional image caption evaluation. In Leibe, B., Matas, J., Sebe, N., and Welling, M., editors, *Computer Vision – ECCV 2016*, pages 382–398, Cham. Springer International Publishing.

Ao, J., Wang, R., Zhou, L., Liu, S., Ren, S., Wu, Y., Ko, T., Li, Q., Zhang, Y., Wei, Z., et al. (2021). SpeechT5: Unified-modal encoder-decoder pre-training for spoken language processing. *arXiv preprint arXiv:2110.07205*.

Ao, J., Wang, R., Zhou, L., Wang, C., Ren, S., Wu, Y., Liu, S., Ko, T., Li, Q., Zhang, Y., Wei, Z., Qian, Y., Li, J., and Wei, F. (2022a). SpeechT5: Unified-modal encoder-decoder pre-training for spoken language processing. In *Proc. ACL*, pages 5723–5738. Association for Computational Linguistics.

Ao, J., Wang, Y., Tian, X., Chen, D., Zhang, J., Lu, L., Wang, Y., Li, H., and Wu, Z. (2024). SD-eval: A benchmark dataset for spoken dialogue understanding beyond words. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Ao, J., Zhang, Z., Zhou, L., Liu, S., Li, H., Ko, T., Dai, L., Li, J., Qian, Y., and Wei, F. (2022b). Pre-Training Transformer Decoder for End-to-End ASR Model with Unpaired Speech Data. In *Proc. Interspeech*, pages 2658–2662.

Ardila, R., Branson, M., Davis, K., Kohler, M., Meyer, J., Henretty, M., Morais, R., Saunders, L., Tyers, F., and Weber, G. (2020). Common voice: A massively-multilingual speech corpus. In *Proceedings of the Twelfth Language Resources and Evaluation*

Conference, pages 4218–4222, Marseille, France. European Language Resources Association.

Baevski, A., Auli, M., and Mohamed, A. (2019). Effectiveness of self-supervised pre-training for speech recognition. *arXiv preprint arXiv:1911.03912*.

Baevski, A., Hsu, W.-N., Xu, Q., Babu, A., Gu, J., and Auli, M. (2022). data2vec: A general framework for self-supervised learning in speech, vision and language. In *Proc. ICML*, volume 162 of *Proceedings of Machine Learning Research*, pages 1298–1312. PMLR.

Baevski, A. and Mohamed, A. (2020). Effectiveness of self-supervised pre-training for asr. In *Proc. ICASSP*, pages 7694–7698.

Baevski, A., Schneider, S., and Auli, M. (2020a). vq-wav2vec: Self-supervised learning of discrete speech representations. In *Proc. ICLR*.

Baevski, A., Zhou, Y., Mohamed, A., and Auli, M. (2020b). wav2vec 2.0: A framework for self-supervised learning of speech representations. In *Proc. NeurIPS*, pages 12449–12460.

Bando, Y., Aizawa, T., Itoyama, K., and Nakadai, K. (2022). Weakly-supervised neural full-rank spatial covariance analysis for a front-end system of distant speech recognition. In *Interspeech 2022*, pages 3824–3828.

Bando, Y., Nakamura, T., and Watanabe, S. (2024). Neural blind source separation and diarization for distant speech recognition. In *Interspeech 2024*, pages 722–726.

Bando, Y., Sekiguchi, K., Masuyama, Y., Nugraha, A. A., Fontaine, M., and Yoshii, K. (2021). Neural full-rank spatial covariance analysis for blind source separation. *IEEE Signal Processing Letters*, 28:1670–1674.

Banerjee, S. and Lavie, A. (2005). METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Bapna, A., Chung, Y.-a., Wu, N., Gulati, A., Jia, Y., Clark, J. H., Johnson, M., Riesa, J., Conneau, A., and Zhang, Y. (2021). Slam: A unified encoder for speech and language modeling via speech-text joint pre-training. *arXiv preprint arXiv:2110.10329*.

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 2018*, pages 1561–1565.

Berant, J., Chou, A., Frostig, R., and Liang, P. (2013). Semantic parsing on Freebase from question-answer pairs. In Yarowsky, D., Baldwin, T., Korhonen, A., Livescu, K., and Bethard, S., editors, *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544, Seattle, Washington, USA. Association for Computational Linguistics.

Boeddeker, C., Subramanian, A. S., Wichern, G., Haeb-Umbach, R., and Le Roux, J. (2024). TS-SEP: Joint diarization and separation conditioned on estimated speaker embeddings. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 32:1185–1197.

Borsdorf, M., Xu, C., Li, H., and Schultz, T. (2021). Universal speaker extraction in the presence and absence of target speakers for speech of one and two talkers. In *Proc. Interspeech 2021*, pages 1469–1473.

Busso, C., Bulut, M., Lee, C.-C., Kazemzadeh, A., Mower, E., Kim, S., Chang, J. N., Lee, S., and Narayanan, S. S. (2008). Iemocap: Interactive emotional dyadic motion capture database. *Language resources and evaluation*, 42:335–359.

Cai, Z., Cao, M., Chen, H., Chen, K., Chen, K., Chen, X., Chen, X., Chen, Z., Chen, Z., Chu, P., et al. (2024). Internlm2 technical report. *arXiv preprint arXiv:2403.17297*.

Cao, H., Cooper, D. G., Keutmann, M. K., Gur, R. C., Nenkova, A., and Verma, R. (2014). Crema-d: Crowd-sourced emotional multimodal actors dataset. *IEEE Transactions on Affective Computing*, 5(4):377–390.

Chen, H., Xie, W., Vedaldi, A., and Zisserman, A. (2020a). Vggsound: A large-scale audio-visual dataset. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 721–725. IEEE.

Chen, Q., Chen, Y., Chen, Y., Chen, M., Chen, Y., Deng, C., Du, Z., Gao, R., Gao, C., Gao, Z., et al. (2025). Minmo: A multimodal large language model for seamless voice interaction. *arXiv preprint arXiv:2501.06282*.

Chen, S., Wang, C., Chen, Z., Wu, Y., Liu, S., Chen, Z., Li, J., Kanda, N., Yoshioka, T., Xiao, X., et al. (2022a). Wavlm: Large-scale self-supervised pre-training for full stack speech processing. *IEEE Journal of Selected Topics in Signal Processing*.

Chen, S., Wang, C., Chen, Z., Wu, Y., Liu, S., Chen, Z., Li, J., Kanda, N., Yoshioka, T., Xiao, X., et al. (2022b). 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, S., Wu, Y., Wang, C., Liu, S., Tompkins, D., Chen, Z., and Wei, F. (2022c). Beats: Audio pre-training with acoustic tokenizers. *arXiv preprint arXiv:2212.09058*.

Chen, X., Zhang, S., Bai, Q., Chen, K., and Nakamura, S. (2024a). LLaST: Improved end-to-end speech translation system leveraged by large language models. In Ku, L.-W., Martins, A., and Srikumar, V., editors, *Findings of the Association for Computational Linguistics: ACL 2024*, pages 6976–6987, Bangkok, Thailand. Association for Computational Linguistics.

Chen, Y., Yue, X., Gao, X., Zhang, C., D'Haro, L. F., Tan, R. T., and Li, H. (2024b). Beyond single-audio: Advancing multi-audio processing in audio large language models. In Al-Onaizan, Y., Bansal, M., and Chen, Y.-N., editors, *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 10917–10930, Miami, Florida, USA. Association for Computational Linguistics.

Chen, Y., Yue, X., Zhang, C., Gao, X., Tan, R. T., and Li, H. (2024c). Voicebench: Benchmarking llm-based voice assistants. *arXiv preprint arXiv:2410.17196*.

Chen, Z., Chen, S., Wu, Y., Qian, Y., Wang, C., Liu, S., Qian, Y., and Zeng, M. (2022d). Large-scale self-supervised speech representation learning for automatic speaker verification. In *Proc. ICASSP*, pages 6147–6151. IEEE.

Chen, Z., Han, B., Wang, S., and Qian, Y. (2023). Attention-based encoder-decoder network for end-to-end neural speaker diarization with target speaker attractor. In *Proc. INTERSPEECH 2023*, pages 3552–3556.

Chen, Z., Han, B., Wang, S., and Qian, Y. (2024d). Attention-based encoder-decoder end-to-end neural diarization with embedding enhancer. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 32:1636–1649.

Chen, Z., Yoshioka, T., Lu, L., Zhou, T., Meng, Z., Luo, Y., Wu, J., Xiao, X., and Li, J. (2020b). Continuous speech separation: Dataset and analysis. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7284–7288. IEEE.

Cheng, X., Dong, Q., Yue, F., Ko, T., Wang, M., and Zou, Y. (2022). M3st: Mix at three levels for speech translation. *arXiv preprint arXiv:2212.03657*.

Cherry, E. C. (1953). Some experiments on the recognition of speech, with one and with two ears. *The Journal of the acoustical society of America*, 25(5):975–979.

Chiang, C.-H., Wang, X., Li, L., Lin, C.-C., Lin, K., Liu, S., Wang, Z., Yang, Z., Lee, H.-y., and Wang, L. (2025). Stitch: Simultaneous thinking and talking with chunked reasoning for spoken language models. *arXiv preprint arXiv:2507.15375*.

Chu, Y., Xu, J., Yang, Q., Wei, H., Wei, X., Guo, Z., Leng, Y., Lv, Y., He, J., Lin, J., et al. (2024). Qwen2-audio technical report. *arXiv preprint arXiv:2407.10759*.

Chu, Y., Xu, J., Zhou, X., Yang, Q., Zhang, S., Yan, Z., Zhou, C., and Zhou, J. (2023). Qwen-audio: Advancing universal audio understanding via unified large-scale audio-language models. *arXiv preprint arXiv:2311.07919*.

Chung, J. S., Huh, J., Mun, S., Lee, M., Heo, H.-S., Choe, S., Ham, C., Jung, S., Lee, B.-J., and Han, I. (2020). In defence of metric learning for speaker recognition. In *Interspeech 2020*, pages 2977–2981.

Chung, J. S., Nagrani, A., and Zisserman, A. (2018). VoxCeleb2: Deep speaker recognition. In *Interspeech 2018*, pages 1086–1090.

Chung, Y.-A. and Glass, J. (2020). Generative pre-training for speech with autoregressive predictive coding. In *Proc. ICASSP*, pages 3497–3501. IEEE.

Clark, H. H. and Fox Tree, J. E. (2002). Using uh and um in spontaneous speaking. *Cognition*, 84(1):73–111.

Cohen, P. R. and Oviatt, S. L. (1995). The role of voice input for human-machine communication. *Proceedings of the National Academy of Sciences*, 92(22):9921–9927.

Contributors, X. (2023). Xtuner: A toolkit for efficiently fine-tuning llm. https://github.com/InternLM/xtuner.

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*.

Cowie, R., Douglas-Cowie, E., Tsapatsoulis, N., Votsis, G., Kollias, S., Fellenz, W., and Taylor, J. (2001). Emotion recognition in human-computer interaction. *IEEE Signal Processing Magazine*, 18(1):32–80.

Défossez, A., Mazaré, L., Orsini, M., Royer, A., Pérez, P., Jégou, H., Grave, E., and Zeghidour, N. (2024). Moshi: a speech-text foundation model for real-time dialogue. *arXiv preprint arXiv:2410.00037*.

Delcroix, M., Zmolikova, K., Kinoshita, K., Ogawa, A., and Nakatani, T. (2018). Single channel target speaker extraction and recognition with speaker beam. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5554–5558.

Delcroix, M., Zmolikova, K., Ochiai, T., Kinoshita, K., Araki, S., and Nakatani, T. (2019). Compact network for SpeakerBeam target speaker extraction. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6965–6969.

Demirsahin, I., Kjartansson, O., Gutkin, A., and Rivera, C. (2020). Open-source Multi-speaker Corpora of the English Accents in the British Isles. In *Proceedings of The 12th Language Resources and Evaluation Conference (LREC)*, pages 6532–6541, Marseille, France. European Language Resources Association (ELRA).

Deshmukh, S., Elizalde, B., Singh, R., and Wang, H. (2023). Pengi: An audio language model for audio tasks. In Oh, A., Neumann, T., Globerson, A., Saenko, K., Hardt, M., and Levine, S., editors, *Advances in Neural Information Processing Systems*, volume 36, pages 18090–18108. Curran Associates, Inc.

Desplanques, B., Thienpondt, J., and Demuynck, K. (2020). ECAPA-TDNN: Emphasized channel attention, propagation and aggregation in TDNN based speaker verification. In *Interspeech 2020*, pages 3830–3834.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proc. NAACL*, pages 4171–4186. Association for Computational Linguistics.

Ding, D., Ju, Z., Leng, Y., Liu, S., Liu, T., Shang, Z., Shen, K., Song, W., Tan, X., Tang, H., et al. (2025). Kimi-audio technical report. *arXiv preprint arXiv:2504.18425*.

Drossos, K., Lipping, S., and Virtanen, T. (2020). Clotho: an audio captioning dataset. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 736–740.

Du, Z., Wang, Y., Chen, Q., Shi, X., Lv, X., Zhao, T., Gao, Z., Yang, Y., Gao, C., Wang, H., et al. (2024). Cosyvoice 2: Scalable streaming speech synthesis with large language models. *arXiv preprint arXiv:2412.10117*.

Ekman, P. and Friesen, W. V. (1971). Constants across cultures in the face and emotion. *Journal of personality and social psychology*, 17(2):124.

Ericsson, K. A. and Simon, H. A. (1984). *Protocol analysis: Verbal reports as data*. MIT press.

Ericsson, L., Gouk, H., Loy, C. C., and Hospedales, T. M. (2022). Self-supervised representation learning: Introduction, advances, and challenges. *IEEE Signal Processing Magazine*, 39(3):42–62.

Fan, Z., Li, M., Zhou, S., and Xu, B. (2021). Exploring wav2vec 2.0 on Speaker Verification and Language Identification. In *Proc. Interspeech 2021*, pages 1509–1513.

Fan, Z., Zhou, S., and Xu, B. (2019). Unsupervised pre-training for sequence to sequence speech recognition. *arXiv preprint arXiv:1910.12418*.

Fang, Q., Guo, S., Zhou, Y., Ma, Z., Zhang, S., and Feng, Y. (2024). Llama-omni: Seamless speech interaction with large language models. *arXiv preprint arXiv:2409.06666*.

Fang, Q., Zhou, Y., Guo, S., Zhang, S., and Feng, Y. (2025). Llama-omni2: Llm-based real-time spoken chatbot with autoregressive streaming speech synthesis. *arXiv preprint arXiv:2505.02625*.

Fazel-Zarandi, M. and Hsu, W.-N. (2023). Cocktail hubert: Generalized self-supervised pre-training for mixture and single-source speech. In *Proc. ICASSP*, pages 1–5. IEEE.

Fonseca, E., Favory, X., Pons, J., Font, F., and Serra, X. (2021). Fsd50k: an open dataset of human-labeled sound events. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:829–852.

Font, F., Roma, G., and Serra, X. (2013). Freesound technical demo. In *Proceedings of the 21st ACM International Conference on Multimedia*, MM '13, pages 411–412, New York, NY, USA. Association for Computing Machinery.

Fu, J., Ng, S.-K., Jiang, Z., and Liu, P. (2023). Gptscore: Evaluate as you desire. *arXiv preprint arXiv:2302.04166*.

Fujita, Y., Kanda, N., Horiguchi, S., Nagamatsu, K., and Watanabe, S. (2019). End-to-end neural speaker diarization with permutation-free objectives. In *Proc. Interspeech 2019*, pages 4300–4304.

Fujita, Y., Watanabe, S., Horiguchi, S., Xue, Y., Shi, J., and Nagamatsu, K. (2020). Neural speaker diarization with speaker-wise chain rule. *arXiv preprint arXiv:2006.01796*.

Gao, C., Cheng, G., Yang, R., Zhu, H., Zhang, P., and Yan, Y. (2021). Pre-training transformer decoder for end-to-end asr model with unpaired text data. In *Proc. ICASSP*, pages 6543–6547. IEEE.

Gao, H., Shao, H., Wang, X., Qiu, C., Shen, Y., Cai, S., Shi, Y., Xu, Z., Long, Z., Zhang, Y., et al. (2025). Lucy: Linguistic understanding and control yielding early stage of her. *arXiv preprint arXiv:2501.16327*.

Ge, M., Xu, C., Wang, L., Chng, E. S., Dang, J., and Li, H. (2020). SpEx+: A complete time domain speaker extraction network. In *Proc. Interspeech 2020*, pages 1406–1410.

Ge, M., Xu, C., Wang, L., Chng, E. S., Dang, J., and Li, H. (2022). L-SpEx: Localized target speaker extraction. In *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7287–7291.

Gemmeke, J. F., Ellis, D. P. W., Freedman, D., Jansen, A., Lawrence, W., Moore, R. C., Plakal, M., and Ritter, M. (2017). Audio set: An ontology and human-labeled dataset for audio events. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 776–780.

Gong, Y., Khurana, S., Karlinsky, L., and Glass, J. (2023a). Whisper-at: Noise-robust automatic speech recognizers are also strong general audio event taggers. *arXiv preprint arXiv:2307.03183*.

Gong, Y., Liu, A. H., Luo, H., Karlinsky, L., and Glass, J. (2023b). Joint audio and speech understanding. In *2023 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 1–8. IEEE.

Gong, Y., Luo, H., Liu, A. H., Karlinsky, L., and Glass, J. R. (2024). Listen, think, and understand. In *The Twelfth International Conference on Learning Representations*.

Gorman, K., Howell, J., and Wagner, M. (2011). Prosodylab-aligner: A tool for forced alignment of laboratory speech. *Canadian Acoustics*, 39(3):192–193.

Graves, A., Fernández, S., Gomez, F., and Schmidhuber, J. (2006). Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proc. Int. Conf. Learn. Representations*, pages 369–376.

Gulcehre, C., Firat, O., Xu, K., Cho, K., Barrault, L., Lin, H.-C., Bougares, F., Schwenk, H., and Bengio, Y. (2015). On using monolingual corpora in neural machine translation. *arXiv preprint arXiv:1503.03535*.

Guo, D., Yang, D., Zhang, H., Song, J., Zhang, R., Xu, R., Zhu, Q., Ma, S., Wang, P., Bi, X., et al. (2025). Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

He, H., Shang, Z., Wang, C., Li, X., Gu, Y., Hua, H., Liu, L., Yang, C., Li, J., Shi, P., et al. (2024). Emilia: An extensive, multilingual, and diverse speech dataset for large-scale speech generation. *arXiv preprint arXiv:2407.05361*.

He, Kaiming and Zhang, Xiangyu and Ren, Shaoqing and Sun, Jian (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Hendrycks, D. and Gimpel, K. (2016). Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*.

Hershey, S., Ellis, D. P. W., Fonseca, E., Jansen, A., Liu, C., Channing Moore, R., and Plakal, M. (2021). The benefit of temporally-strong labels in audio event classification. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 366–370.

Ho, J. and Salimans, T. (2022). Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*.

Hori, T., Watanabe, S., and Hershey, J. (2017). Joint CTC/attention decoding for end-to-end speech recognition. In *Proc. ACL*, pages 518–529.

Horiguchi, S., Fujita, Y., Watanabe, S., Xue, Y., and Nagamatsu, K. (2020). End-to-end speaker diarization for an unknown number of speakers with encoder-decoder based attractors. In *Proc. Interspeech 2020*, pages 269–273.

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, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. (2021). Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Hu, S., Zhou, L., Liu, S., Chen, S., Hao, H., Pan, J., Liu, X., Li, J., Sivasankaran, S., Liu, L., et al. (2024). Wavllm: Towards robust and adaptive speech large language model. *arXiv preprint arXiv:2404.00656*.

Huang, C.-y., Chen, W.-C., Yang, S.-w., Liu, A. T., Li, C.-A., Lin, Y.-X., Tseng, W.-C., Diwan, A., Shih, Y.-J., Shi, J., et al. (2024a). Dynamic-superb phase-2: A collaboratively expanding benchmark for measuring the capabilities of spoken language models with 180 tasks. *arXiv preprint arXiv:2411.05361*.

Huang, C.-y., Lu, K.-H., Wang, S.-H., Hsiao, C.-Y., Kuan, C.-Y., Wu, H., Arora, S., Chang, K.-W., Shi, J., Peng, Y., et al. (2024b). Dynamic-superb: Towards a dynamic, collaborative, and comprehensive instruction-tuning benchmark for speech. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 12136–12140. IEEE.

Huang, Z., Raj, D., García, P., and Khudanpur, S. (2023). Adapting self-supervised models to multi-talker speech recognition using speaker embeddings. In *Proc. ICASSP*, pages 1–5. IEEE.

Hurst, A., Lerer, A., Goucher, A. P., Perelman, A., Ramesh, A., Clark, A., Ostrow, A., Welihinda, A., Hayes, A., Radford, A., et al. (2024). Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Jaech, A., Kalai, A., Lerer, A., Richardson, A., El-Kishky, A., Low, A., Helyar, A., Madry, A., Beutel, A., Carney, A., et al. (2024). Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

James, J., Tian, L., and Inez Watson, C. (2018). An Open Source Emotional Speech Corpus for Human Robot Interaction Applications. In *Proc. Interspeech 2018*, pages 2768–2772.

Jiang, D., Li, W., Zhang, R., Cao, M., Luo, N., Han, Y., Zou, W., Han, K., and Li, X. (2021). A further study of unsupervised pretraining for transformer based speech recognition. In *Proc. ICASSP*, pages 6538–6542. IEEE.

Jiang, Y., Chen, Z., Tao, R., Deng, L., Qian, Y., and Li, H. (2024). Prompt-driven target speech diarization. In *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 11086–11090.

Joshi, M., Choi, E., Weld, D. S., and Zettlemoyer, L. (2017). Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*.

Ju, Z., Wang, Y., Shen, K., Tan, X., Xin, D., Yang, D., Liu, Y., Leng, Y., Song, K., Tang, S., et al. (2024). Naturalspeech 3: Zero-shot speech synthesis with factorized codec and diffusion models. *arXiv preprint arXiv:2403.03100*.

Kahn, J., Riviere, M., Zheng, W., Kharitonov, E., Xu, Q., Mazaré, P.-E., Karadayi, J., Liptchinsky, V., Collobert, R., Fuegen, C., et al. (2020). Libri-Light: A benchmark for ASR with limited or no supervision. In *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, pages 7669–7673.

Kalda, J., Pagés, C., Marxer, R., Alumäe, T., and Bredin, H. (2024). PixIT: Joint training of speaker diarization and speech separation from real-world multi-speaker recordings. In *The Speaker and Language Recognition Workshop (Odyssey 2024)*, pages 115–122.

Kanda, N., Gaur, Y., Wang, X., Meng, Z., Chen, Z., Zhou, T., and Yoshioka, T. (2020). Joint speaker counting, speech recognition, and speaker identification for overlapped speech of any number of speakers. In *Interspeech 2020*, pages 36–40.

Kanda, N., Meng, Z., Lu, L., Gaur, Y., Wang, X., Chen, Z., and Yoshioka, T. (2021a). Minimum Bayes risk training for end-to-end speaker-attributed ASR. In *ICASSP*

2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 6503–6507. IEEE.

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

Kanda, N., Ye, G., Gaur, Y., Wang, X., Meng, Z., Chen, Z., and Yoshioka, T. (2021b). End-to-End Speaker-Attributed ASR with Transformer. In *Proc. Interspeech 2021*, pages 4413–4417.

Kim, C. D., Kim, B., Lee, H., and Kim, G. (2019). Audiocaps: Generating captions for audios in the wild. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 119–132.

Kim, J., Lee, K., Chung, S., and Cho, J. (2024). Clam-tts: Improving neural codec language model for zero-shot text-to-speech. *arXiv preprint arXiv:2404.02781*.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kinoshita, K., Delcroix, M., and Tawara, N. (2021). Advances in integration of end-to-end neural and clustering-based diarization for real conversational speech. In *Interspeech 2021*, pages 3565–3569.

Kinoshita, Keisuke and Delcroix, Marc and Tawara, Naohiro (2021). Integrating end-to-end neural and clustering-based diarization: Getting the best of both worlds. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7198–7202.

Kong, Z., Goel, A., Badlani, R., Ping, W., Valle, R., and Catanzaro, B. (2024). Audio flamingo: A novel audio language model with few-shot learning and dialogue abilities. *arXiv preprint arXiv:2402.01831*.

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. IEEE.

Łajszczak, M., Cámbara, G., Li, Y., Beyhan, F., van Korlaar, A., Yang, F., Joly, A., Martín-Cortinas, Á., Abbas, A., Michalski, A., et al. (2024). Base tts: Lessons from building a billion-parameter text-to-speech model on 100k hours of data. *arXiv preprint arXiv:2402.08093*.

Lakhotia, K., Kharitonov, E., Hsu, W.-N., Adi, Y., Polyak, A., Bolte, B., Nguyen, T.-A., Copet, J., Baevski, A., Mohamed, A., and Dupoux, E. (2021). On generative spoken language modeling from raw audio. *Transactions of the Association for Computational Linguistics*, 9:1336–1354.

Lee, A., Chen, P.-J., Wang, C., Gu, J., Ma, X., Polyak, A., Adi, Y., He, Q., Tang, Y., Pino, J., et al. (2021). Direct speech-to-speech translation with discrete units. *arXiv preprint arXiv:2107.05604*.

Levelt, W. J. (1993). *Speaking: From intention to articulation*. MIT press.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. (2020). BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proc. ACL*, pages 7871–7880. Association for Computational Linguistics.

Li, C.-H., Wu, S.-L., Liu, C.-L., and Lee, H.-y. (2018). Spoken squad: A study of mitigating the impact of speech recognition errors on listening comprehension. *arXiv preprint arXiv:1804.00320*.

Li, G., Liu, J., Dinkel, H., Niu, Y., Zhang, J., and Luan, J. (2025a). Reinforcement learning outperforms supervised fine-tuning: A case study on audio question answering. *arXiv preprint arXiv:2503.11197*.

Li, J. (2021). Recent advances in end-to-end automatic speech recognition. *arXiv preprint arXiv:2111.01690*.

Li, K., Yang, R., and Hu, X. (2023). An efficient encoder-decoder architecture with top-down attention for speech separation. In *The Eleventh International Conference on Learning Representations*.

Li, T., Liu, J., Zhang, T., Fang, Y., Pan, D., Wang, M., Liang, Z., Li, Z., Lin, M., Dong, G., et al. (2025b). Baichuan-audio: A unified framework for end-to-end speech interaction. *arXiv preprint arXiv:2502.17239*.

Lin, C.-Y. (2004). ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Lin, G.-T., Chiang, C.-H., and Lee, H.-y. (2024). Advancing large language models to capture varied speaking styles and respond properly in spoken conversations. *arXiv preprint arXiv:2402.12786*.

Lin, J., Yue, X., Ao, J., and Li, H. (2023). Self-Supervised Acoustic Word Embedding Learning via Correspondence Transformer Encoder. In *Proc. INTERSPEECH 2023*, pages 2988–2992.

Lipping, S., Sudarsanam, P., Drossos, K., and Virtanen, T. (2022). Clotho-aqa: A crowdsourced dataset for audio question answering. In *2022 30th European Signal Processing Conference (EUSIPCO)*, pages 1140–1144. IEEE.

Liu, H., Li, C., Wu, Q., and Lee, Y. J. (2024a). Visual instruction tuning. *Advances in neural information processing systems*, 36.

Liu, K., Du, Z., Wan, X., and Zhou, H. (2023a). X-SepFormer: End-to-end speaker extraction network with explicit optimization on speaker confusion. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.

Liu, S., Hussain, A. S., Sun, C., and Shan, Y. (2024b). Music understanding llama: Advancing text-to-music generation with question answering and captioning. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 286–290. IEEE.

Liu, S., Zhu, Z., Ye, N., Guadarrama, S., and Murphy, K. (2017). Improved image captioning via policy gradient optimization of spider. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

Liu, T., Lee, K. A., Wang, Q., and Li, H. (2023b). Disentangling voice and content with self-supervision for speaker recognition. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pages 50221–50236.

Liu, Y., Iter, D., Xu, Y., Wang, S., Xu, R., and Zhu, C. (2023c). G-eval: NLG evaluation using gpt-4 with better human alignment. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2511–2522, Singapore. Association for Computational Linguistics.

Long, Z., Shen, Y., Fu, C., Gao, H., Li, L., Chen, P., Zhang, M., Shao, H., Li, J., Peng, J., et al. (2025). Vita-audio: Fast interleaved cross-modal token generation for efficient large speech-language model. *arXiv preprint arXiv:2505.03739*.

Loshchilov, I. and Hutter, F. (2017). Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Lotfian, R. and Busso, C. (2019). Building naturalistic emotionally balanced speech corpus by retrieving emotional speech from existing podcast recordings. *IEEE Transactions on Affective Computing*, 10(4):471–483.

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, 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.

Luo, Y. and Yu, J. (2023). Music source separation with band-split RNN. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31:1893–1901.

Ma, Z., Zheng, Z., Tang, C., Wang, Y., and Chen, X. (2023a). MT4SSL: Boosting Self-Supervised Speech Representation Learning by Integrating Multiple Targets. *Proc. of InterSpeech*.

Ma, Z., Zheng, Z., Ye, J., Li, J., Gao, Z., Zhang, S., and Chen, X. (2023b). emotion2vec: Self-supervised pre-training for speech emotion representation. *arXiv preprint arXiv:2312.15185*.

Maiti, S., Ueda, Y., Watanabe, S., Zhang, C., Yu, M., Zhang, S.-X., and Xu, Y. (2023). EEND-SS: Joint end-to-end neural speaker diarization and speech separation for flexible number of speakers. In *2022 IEEE Spoken Language Technology Workshop (SLT)*, pages 480–487.

Mansfield, C., Ng, S., Levow, G.-A., Wright, R. A., and Ostendorf, M. (2021). Revisiting parity of human vs. machine conversational speech transcription. In *Interspeech 2021*, pages 1997–2001.

Mark, P. and Martin, A. (2001). 2000 NIST speaker recognition evaluation.

McAuliffe, M., Socolof, M., Mihuc, S., Wagner, M., and Sonderegger, M. (2017). Montreal Forced Aligner: Trainable Text-Speech Alignment Using Kaldi. In *Proc. Interspeech 2017*, pages 498–502.

McAuliffe, M. and Sonderegger, M. (2022). English (us) arpa acoustic model v2.0.0a. Technical report, https://mfa-models.readthedocs.io/acoustic/English/English(US)ARPAacousticmodelv2_0_0a.html.

McKeown, G., Valstar, M. F., Cowie, R., and Pantic, M. (2010). The semaine corpus of emotionally coloured character interactions. In *2010 IEEE international conference on multimedia and expo*, pages 1079–1084. IEEE.

Medennikov, I., Korenevsky, M., Prisyach, T., Khokhlov, Y., Korenevskaya, M., Sorokin, I., Timofeeva, T., Mitrofanov, A., Andrusenko, A., Podluzhny, I., Laptev, A., and Romanenko, A. (2020). Target-Speaker Voice Activity Detection: A Novel Approach for Multi-Speaker Diarization in a Dinner Party Scenario. In *Proc. Interspeech 2020*, pages 274–278.

Mohamed, A., Lee, H.-y., Borgholt, L., Havtorn, J. D., Edin, J., Igel, C., Kirchhoff, K., Li, S.-W., Livescu, K., Maaløe, L., et al. (2022). Self-supervised speech representation learning: A review. *IEEE Journal of Selected Topics in Signal Processing*, 16(6):1179–1210.

Morais, E., Hoory, R., Zhu, W., Gat, I., Damasceno, M., and Aronowitz, H. (2022). Speech emotion recognition using self-supervised features. In *Proc. ICASSP*, pages 6922–6926. IEEE.

Nachmani, E., Levkovitch, A., Hirsch, R., Salazar, J., Asawaroengchai, C., Mariooryad, S., Rivlin, E., Skerry-Ryan, R., and Ramanovich, M. T. (2023). Spoken question answering and speech continuation using spectrogram-powered llm. In *The Twelfth International Conference on Learning Representations*.

Nass, C. and Brave, S. (2005). *Wired for Speech: How Voice Activates and Advances the Human-Computer Relationship*. The MIT Press.

Newell, A. and Simon, H. A. (1972). *Human problem solving*. Prentice-Hall.

Novikova, J., Dušek, O., Curry, A. C., and Rieser, V. (2017). Why we need new evaluation metrics for nlg. *arXiv preprint arXiv:1707.06875*.

Oord, A. v. d., Li, Y., and Vinyals, O. (2018). Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

OpenAI (2024). Gpt-4o. https://openai.com/index/hello-gpt-4o/.

Ott, M., Edunov, S., Baevski, A., Fan, A., Gross, S., Ng, N., Grangier, D., and Auli, M. (2019). fairseq: A fast, extensible toolkit for sequence modeling. In *Proc. NAACL-HLT*.

Pan, Z., Ge, M., and Li, H. (2022). USEV: Universal speaker extraction with visual cue. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:3032–3045.

Panayotov, V., Chen, G., Povey, D., and Khudanpur, S. (2015a). Librispeech: an asr corpus based on public domain audio books. In *Proc. ICASSP*, pages 5206–5210. IEEE.

Panayotov, V., Chen, G., Povey, D., and Khudanpur, S. (2015b). Librispeech: An asr corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Park, T. J., Kanda, N., Dimitriadis, D., Han, K. J., Watanabe, S., and Narayanan, S. (2022). A review of speaker diarization: Recent advances with deep learning. *Computer Speech & Language*, 72:101317.

Peng, P., Huang, P.-Y., Li, D., Mohamed, A., and Harwath, D. (2024). Voicecraft: Zero-shot speech editing and text-to-speech in the wild. *arXiv preprint arXiv:2403.16973*.

Poria, S., Hazarika, D., Majumder, N., Naik, G., Cambria, E., and Mihalcea, R. (2019). MELD: A multimodal multi-party dataset for emotion recognition in conversations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 527–536, Florence, Italy. Association for Computational Linguistics.

Pradhan, S., Cole, R. A., and Ward, W. H. (2024). My science tutor (MyST)–a large corpus of children's conversational speech. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 12040–12045, Torino, Italia. ELRA and ICCL.

Pratap, V., Hannun, A., Xu, Q., Cai, J., Kahn, J., Synnaeve, G., Liptchinsky, V., and Collobert, R. (2019). Wav2letter++: A fast open-source speech recognition system. In *Proc. ICASSP*, pages 6460–6464.

Qian, Y.-m., Weng, C., Chang, X.-k., Wang, S., and Yu, D. (2018). Past review, current progress, and challenges ahead on the cocktail party problem. *Frontiers of Information Technology & Electronic Engineering*, 19:40–63.

Radford, A., Kim, J. W., Xu, T., Brockman, G., McLeavey, C., and Sutskever, I. (2023). Robust speech recognition via large-scale weak supervision. In *International conference on machine learning*, pages 28492–28518. PMLR.

Rafailov, R., Sharma, A., Mitchell, E., Manning, C. D., Ermon, S., and Finn, C. (2024). Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36.

Rao, W., Xu, C., Chng, E. S., and Li, H. (2019). Target speaker extraction for multi-talker speaker verification. In *Proc. Interspeech 2019*, pages 1273–1277.

Reddy, C. K., 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. IEEE.

Reddy, C. K., Gopal, V., Cutler, R., Beyrami, E., Cheng, R., Dubey, H., Matusevych, S., Aichner, R., Aazami, A., Braun, S., Rana, P., Srinivasan, S., and Gehrke, J. (2020). The INTERSPEECH 2020 Deep Noise Suppression Challenge: Datasets, Subjective Testing Framework, and Challenge Results. In *Proc. Interspeech 2020*, pages 2492–2496.

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 vol.2.

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.

Sailunaz, K. and Alhajj, R. (2019). Emotion and sentiment analysis from twitter text. *Journal of computational science*, 36:101003.

Sakshi, S., Tyagi, U., Kumar, S., Seth, A., Selvakumar, R., Nieto, O., Duraiswami, R., Ghosh, S., and Manocha, D. (2024). Mmau: A massive multi-task audio understanding and reasoning benchmark. *arXiv preprint arXiv:2410.19168*.

Sanchez, G., Fan, H., Spangher, A., Levi, E., Ammanamanchi, P. S., and Biderman, S. (2023). Stay on topic with classifier-free guidance. *arXiv preprint arXiv:2306.17806*.

Schneider, S., Baevski, A., Collobert, R., and Auli, M. (2019). wav2vec: Unsupervised Pre-Training for Speech Recognition. In *Proc. Interspeech*, pages 3465–3469.

Sell, G. and Garcia-Romero, D. (2014). Speaker diarization with PLDA i-vector scoring and unsupervised calibration. In *2014 IEEE Spoken Language Technology Workshop (SLT)*, pages 413–417.

Senoussaoui, M., Kenny, P., Stafylakis, T., and Dumouchel, P. (2014). A study of the cosine distance-based mean shift for telephone speech diarization. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 22(1):217–227.

Series, B. (2011). Algorithms to measure audio programme loudness and true-peak audio level. In *International Telecommunication Union Radiocommunication Assembly*.

Shafey, L. E., Soltau, H., and Shafran, I. (2019). Joint speech recognition and speaker diarization via sequence transduction. In *Interspeech 2019*, pages 396–400.

Shao, Z., Luo, Y., Lu, C., Ren, Z., Hu, J., Ye, T., Gou, Z., Ma, S., and Zhang, X. (2025). Deepseekmath-v2: Towards self-verifiable mathematical reasoning.

Shaw, P., Uszkoreit, J., and Vaswani, A. (2018). Self-attention with relative position representations. In *Proc. NAACL-HLT*, pages 464–468.

Shi, J., Chang, X., Hayashi, T., Lu, Y.-J., Watanabe, S., and Xu, B. (2021). Discretization and re-synthesis: an alternative method to solve the cocktail party problem. *arXiv preprint arXiv:2112.09382*.

Shih, Y.-J., Raj, D., Wu, C., Zhou, W., Bong, S., Gaur, Y., Mahadeokar, J., Kalinli, O., and Seltzer, M. L. (2025). Can speech llms think while listening? *ArXiv*, abs/2510.07497.

Shum, S. H., Dehak, N., Dehak, R., and Glass, J. R. (2013). Unsupervised methods for speaker diarization: An integrated and iterative approach. *IEEE Transactions on Audio, Speech, and Language Processing*, 21(10):2015–2028.

SoundBible (2006). Soundbible.

Su, D. and Fung, P. (2020). Improving spoken question answering using contextualized word representation. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8004–8008. IEEE.

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 *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 4214–4217.

Tang, C., Yu, W., Sun, G., Chen, X., Tan, T., Li, W., Lu, L., MA, Z., and Zhang, C. (2024). SALMONN: Towards generic hearing abilities for large language models. In *The Twelfth International Conference on Learning Representations*.

Tang, Y., Gong, H., Dong, N., Wang, C., Hsu, W.-N., Gu, J., Baevski, A., Li, X., Mohamed, A., Auli, M., and Pino, J. (2022). Unified speech-text pre-training for speech translation and recognition. In *Proc. ACL*, pages 1488–1499. Association for Computational Linguistics.

Team, G., Anil, R., Borgeaud, S., Wu, Y., Alayrac, J.-B., Yu, J., Soricut, R., Schalkwyk, J., Dai, A. M., Hauth, A., et al. (2023). Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Team, G., Riviere, M., Pathak, S., Sessa, P. G., Hardin, C., Bhupatiraju, S., Hussenot, L., Mesnard, T., Shahriari, B., Ramé, A., et al. (2024). Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*.

Team, Q. (2025). Qwq-32b: Embracing the power of reinforcement learning.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al. (2023a). Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al. (2023b). Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Tsai, H.-S., Chang, H.-J., Huang, W.-C., Huang, Z., Lakhotia, K., Yang, S.-w., Dong, S., Liu, A., Lai, C.-I., Shi, J., Chang, X., Hall, P., Chen, H.-J., Li, S.-W., Watanabe, S., Mohamed, A., and Lee, H.-y. (2022). SUPERB-SG: Enhanced speech processing universal PERformance benchmark for semantic and generative capabilities. In *Proc. ACL*, pages 8479–8492. Association for Computational Linguistics.

Tseng, B.-H., Shen, S.-S., Lee, H.-Y., and Lee, L.-S. (2016). Towards machine comprehension of spoken content: Initial toefl listening comprehension test by machine. *arXiv preprint arXiv:1608.06378*.

Veaux, C., Yamagishi, J., and King, S. (2013). The voice bank corpus: Design, collection and data analysis of a large regional accent speech database. In *2013 international conference oriental COCOSDA held jointly with 2013 conference on Asian spoken language research and evaluation (O-COCOSDA/CASLRE)*, pages 1–4. IEEE.

Vedantam, R., Lawrence Zitnick, C., and Parikh, D. (2015). Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575.

Wang, B., Zou, X., Lin, G., Sun, S., Liu, Z., Zhang, W., Liu, Z., Aw, A., and Chen, N. F. (2024a). Audiobench: A universal benchmark for audio large language models. *arXiv preprint arXiv:2406.16020*.

Wang, C., Chen, S., Wu, Y., Zhang, Z., Zhou, L., Liu, S., Chen, Z., Liu, Y., Wang, H., Li, J., et al. (2023a). Neural codec language models are zero-shot text to speech synthesizers. *arXiv preprint arXiv:2301.02111*.

Wang, C., Wu, A., and Pino, J. (2020a). Covost 2: A massively multilingual speech-to-text translation corpus.

Wang, C., Wu, Y., Qian, Y., Kumatani, K., Liu, S., Wei, F., Zeng, M., and Huang, X. (2021a). Unispeech: Unified speech representation learning with labeled and unlabeled data. In *Proc. ICML*, pages 10937–10947. PMLR.

Wang, H., Liang, C., Wang, S., Chen, Z., Zhang, B., Xiang, X., Deng, Y., and Qian, Y. (2023b). Wespeaker: A research and production oriented speaker embedding learning toolkit. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.

Wang, H., Zheng, S., Chen, Y., Cheng, L., and Chen, Q. (2023c). Cam++: A fast and efficient network for speaker verification using context-aware masking. In *INTERSPEECH*.

Wang, K., Wu, Q., Song, L., Yang, Z., Wu, W., Qian, C., He, R., Qiao, Y., and Loy, C. C. (2020b). Mead: A large-scale audio-visual dataset for emotional talking-face generation. In *ECCV*.

Wang, Q., Downey, C., Wan, L., Mansfield, P. A., and Moreno, I. L. (2018). Speaker diarization with LSTM. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5239–5243.

Wang, Q., Muckenhirn, H., Wilson, K., Sridhar, P., Wu, Z., Hershey, J. R., Saurous, R. A., Weiss, R. J., Jia, Y., and Moreno, I. L. (2019). VoiceFilter: Targeted voice separation by speaker-conditioned spectrogram masking. In *Proc. Interspeech 2019*, pages 2728–2732.

Wang, S., Chen, Z., Lee, K. A., Qian, Y., and Li, H. (2024b). Overview of speaker modeling and its applications: From the lens of deep speaker representation learning. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 32:4971–4998.

Wang, S., Zhang, K., Lin, S., Li, J., Wang, X., Ge, M., Yu, J., Qian, Y., and Li, H. (2024c). WeSep: A scalable and flexible toolkit towards generalizable target speaker extraction. In *Interspeech 2024*, pages 4273–4277.

Wang, T., Chen, X., Chen, Z., Yu, S., and Zhu, W. (2023d). An adapter based multi-label pre-training for speech separation and enhancement. In *Proc. ICASSP*, pages 1–5. IEEE.

Wang, W., Tang, Q., and Livescu, K. (2020c). Unsupervised pre-training of bidirectional speech encoders via masked reconstruction. In *Proc. ICASSP*, pages 6889–6893. IEEE.

Wang, X., Li, Y., Fu, C., Shen, Y., Xie, L., Li, K., Sun, X., and Ma, L. (2024d). Freeze-omni: A smart and low latency speech-to-speech dialogue model with frozen llm. *arXiv preprint arXiv:2411.00774*.

Wang, Y., Li, J., Wang, H., Qian, Y., Wang, C., and Wu, Y. (2022). Wav2vec-switch: Contrastive learning from original-noisy speech pairs for robust speech recognition. In *Proc. ICASSP*, pages 7097–7101. IEEE.

Wang, Y., Zhan, H., Liu, L., Zeng, R., Guo, H., Zheng, J., Zhang, Q., Zhang, X., Zhang, S., and Wu, Z. (2024e). Maskgct: Zero-shot text-to-speech with masked generative codec transformer. *arXiv preprint arXiv:2409.00750*.

Wang, Y.-X., Du, J., He, M., Niu, S.-T., Sun, L., and Lee, C.-H. (2021b). Scenario-dependent speaker diarization for DIHARD-III challenge. In *Proc. Interspeech 2021*, pages 3106–3110.

Watanabe, S., Hori, T., Kim, S., Hershey, J. R., and Hayashi, T. (2017). Hybrid ctc/attention architecture for end-to-end speech recognition. *IEEE Journal of Selected Topics in Signal Processing*, 11(8):1240–1253.

Watanabe, S., Mandel, M., Barker, J., Vincent, E., Arora, A., Chang, X., Khudanpur, S., Manohar, V., Povey, D., Raj, D., Snyder, D., Subramanian, A. S., Trmal, J., Yair, B. B., Boeddeker, C., Ni, Z., Fujita, Y., Horiguchi, S., Kanda, N., Yoshioka, T., and Ryant, N. (2020). CHiME-6 Challenge: Tackling Multispeaker Speech Recognition for Unsegmented Recordings. In *Proc. 6th International Workshop on Speech Processing in Everyday Environments (CHiME 2020)*, pages 1–7.

Wei, C., Wang, B., Kim, J.-j., and Chen, N. F. (2025). Towards spoken mathematical reasoning: Benchmarking speech-based models over multi-faceted math problems. *arXiv preprint arXiv:2505.15000*.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q. V., Zhou, D., et al. (2022). Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Wen, C., Guo, T., Zhao, S., Zou, W., and Li, X. (2025). Sari: Structured audio reasoning via curriculum-guided reinforcement learning. *arXiv preprint arXiv:2504.15900*.

wen Yang, S., 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., tik Lee, K., Liu, D.-R., Huang, Z., Dong, S., Li, S.-W., Watanabe, S., Mohamed, A., and yi Lee, H. (2021). Superb: Speech processing universal performance benchmark. In *Interspeech 2021*, pages 1194–1198.

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 2019*, pages 1368–1372.

Wisdom, S., Tzinis, E., Erdogan, H., Weiss, R., Wilson, K., and Hershey, J. (2020). Unsupervised sound separation using mixture invariant training. *Advances in neural information processing systems*, 33:3846–3857.

Wu, Y., Rallabandi, S., Srinivasamurthy, R., Dakle, P. P., Gon, A., and Raghavan, P. (2023). Heysquad: A spoken question answering dataset. *arXiv preprint arXiv:2304.13689*.

Xie, Z., Lin, M., Liu, Z., Wu, P., Yan, S., and Miao, C. (2025a). Audio-reasoner: Improving reasoning capability in large audio language models. *arXiv preprint arXiv:2503.02318*.

Xie, Z., Ma, Z., Liu, Z., Pang, K., Li, H., Zhang, J., Liao, Y., Ye, D., Miao, C., and Yan, S. (2025b). Mini-omni-reasoner: Token-level thinking-in-speaking in large speech models. *arXiv preprint arXiv:2508.15827*.

Xie, Z. and Wu, C. (2024). Mini-omni: Language models can hear, talk while thinking in streaming. *arXiv preprint arXiv:2408.16725*.

Xu, C., Rao, W., Chng, E. S., and Li, H. (2020). SpEx: Multi-scale time domain speaker extraction network. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:1370–1384.

Xu, C., Rao, W., Wu, J., and Li, H. (2021). Target speaker verification with selective auditory attention for single and multi-talker speech. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:2696–2709.

Xu, J., Guo, Z., He, J., Hu, H., He, T., Bai, S., Chen, K., Wang, J., Fan, Y., Dang, K., et al. (2025). Qwen2. 5-omni technical report. *arXiv preprint arXiv:2503.20215*.

Xue, H., Liang, Y., Mu, B., Zhang, S., Chen, Q., and Xie, L. (2023). E-chat: Emotion-sensitive spoken dialogue system with large language models. *arXiv preprint arXiv:2401.00475*.

Yamagishi, J., Veaux, C., and MacDonald, K. (2019). CSTR VCTK Corpus: English multi-speaker corpus for CSTR voice cloning toolkit.

Yang, A., Yang, B., Hui, B., Zheng, B., Yu, B., Zhou, C., Li, C., Li, C., Liu, D., Huang, F., et al. (2024a). Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Yang, Q., Xu, J., Liu, W., Chu, Y., Jiang, Z., Zhou, X., Leng, Y., Lv, Y., Zhao, Z., Zhou, C., et al. (2024b). Air-bench: Benchmarking large audio-language models via generative comprehension. *arXiv preprint arXiv:2402.07729*.

Yang, Q. A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Li, C., Liu, D., Huang, F., Dong, G., Wei, H., Lin, H., Yang, J., Tu, J., Zhang, J., Yang, J., Yang, J., Zhou, J., Lin, J., Dang, K., Lu, K., Bao, K., Yang, K., Yu, L., Li, M., Xue, M., Zhang, P., Zhu, Q., Men, R., Lin, R., Li, T., Xia, T., Ren, X., Ren, X., Fan, Y., Su, Y., Zhang, Y.-C., Wan, Y., Liu, Y., Cui, Z., Zhang, Z., Qiu, Z., Quan, S., and Wang, Z. (2024c). Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

Yao, Y., Yu, T., Zhang, A., Wang, C., Cui, J., Zhu, H., Cai, T., Li, H., Zhao, W., He, Z., et al. (2024). Minicpm-v: A gpt-4v level mllm on your phone. *arXiv preprint arXiv:2408.01800*.

Ye, R., Wang, M., and Li, L. (2022). Cross-modal contrastive learning for speech translation. In *Proc. ACL*, pages 5099–5113. Association for Computational Linguistics.

You, C., Chen, N., Liu, F., Ge, S., Wu, X., and Zou, Y. (2022). End-to-end spoken conversational question answering: Task, dataset and model. *arXiv preprint arXiv:2204.14272*.

Young, A., Chen, B., Li, C., Huang, C., Zhang, G., Zhang, G., Li, H., Zhu, J., Chen, J., Chang, J., et al. (2024). Yi: Open foundation models by 01. ai. *arXiv preprint arXiv:2403.04652*.

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 *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 241–245. IEEE.

Yu, J., Chen, H., Luo, Y., Gu, R., and Weng, C. (2023). High Fidelity Speech Enhancement with Band-split RNN. In *Proc. Interspeech 2023*, pages 2483–2487.

Zeinali, H., Wang, S., Silnova, A., Matějka, P., and Plchot, O. (2019). BUT system description to VoxCeleb speaker recognition challenge 2019. *arXiv preprint arXiv:1910.12592*.

Zeng, A., Du, Z., Liu, M., Wang, K., Jiang, S., Zhao, L., Dong, Y., and Tang, J. (2024). Glm-4-voice: Towards intelligent and human-like end-to-end spoken chatbot. *arXiv preprint arXiv:2412.02612*.

Zhang, B., Lv, H., Guo, P., Shao, Q., Yang, C., Xie, L., Xu, X., Bu, H., Chen, X., Zeng, C., et al. (2022). Wenetspeech: A 10000+ hours multi-domain mandarin corpus for speech recognition. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6182–6186. IEEE.

Zhang, D., Li, S., Zhang, X., Zhan, J., Wang, P., Zhou, Y., and Qiu, X. (2023a). SpeechGPT: Empowering large language models with intrinsic cross-modal conversational abilities. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 15757–15773, Singapore. Association for Computational Linguistics.

Zhang, P., Wang, X. D. B., Cao, Y., Xu, C., Ouyang, L., Zhao, Z., Ding, S., Zhang, S., Duan, H., Yan, H., et al. (2023b). Internlm-xcomposer: A vision-language large model for advanced text-image comprehension and composition. *arXiv preprint arXiv:2309.15112*.

Zhang*, T., Kishore*, V., Wu*, F., Weinberger, K. Q., and Artzi, Y. (2020). Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Zhang, W. and Qian, Y. (2023). Weakly-supervised speech pre-training: A case study on target speech recognition. In *Proc. Interspeech 2023*.

Zhang, X., Xue, L., Gu, Y., Wang, Y., Li, J., He, H., Wang, C., Song, T., Chen, X., Fang, Z., Chen, H., Zhang, J., Tang, T. Y., Zou, L., Wang, M., Han, J., Chen, K., Li, H., and Wu, Z. (2024). Amphion: An open-source audio, music and speech generation toolkit. In *IEEE Spoken Language Technology Workshop, SLT 2024*.

Zhang, Z., He, B., and Zhang, Z. (2020). X-TaSNet: Robust and accurate time-domain speaker extraction network. In *Proc. Interspeech 2020*, pages 1421–1425.

Zheng, L., Chiang, W.-L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., Lin, Z., Li, Z., Li, D., Xing, E., et al. (2024). Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36.

Zhou, K., Sisman, B., Liu, R., and Li, H. (2022). Emotional voice conversion: Theory, databases and esd. *Speech Communication*, 137:1–18.

Zhou, T., Zhao, Y., and Wu, J. (2021). ResNeXt and Res2Net structures for speaker verification. In *2021 IEEE Spoken Language Technology Workshop (SLT)*, pages 301–307. IEEE.

Zhu, D., Chen, J., Shen, X., Li, X., and Elhoseiny, M. (2023a). Minigpt-4: Enhancing vision-language understanding with advanced large language models. *arXiv preprint arXiv:2304.10592*.

Zhu, Q.-S., Zhou, L., Zhang, J., Liu, S.-J., Hu, Y.-C., and Dai, L.-R. (2023b). Robust data2vec: Noise-robust speech representation learning for asr by combining regression and improved contrastive learning. In *Proc. ICASSP*, pages 1–5. IEEE.

Žmolíková, K., Delcroix, M., Kinoshita, K., Ochiai, T., Nakatani, T., Burget, L., and Černockỳ, J. (2019). SpeakerBeam: Speaker aware neural network for target speaker extraction in speech mixtures. *IEEE Journal of Selected Topics in Signal Processing*, 13(4):800–814.

Zmolikova, K., Delcroix, M., Ochiai, T., Kinoshita, K., Černockỳ, J., and Yu, D. (2023). Neural target speech extraction: An overview. *IEEE Signal Processing Magazine*, 40(3):8–29.

Özgür Çetin and Shriberg, E. (2006). Analysis of overlaps in meetings by dialog factors, hot spots, speakers, and collection site: insights for automatic speech recognition. In *Proc. Interspeech*.

Žmolíková, K., Delcroix, M., Kinoshita, K., Higuchi, T., Ogawa, A., and Nakatani, T. (2017a). Learning speaker representation for neural network based multichannel speaker extraction. In *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 8–15.

Žmolíková, K., Delcroix, M., Kinoshita, K., Higuchi, T., Ogawa, A., and Nakatani, T. (2017b). Speaker-aware neural network based beamformer for speaker extraction in speech mixtures. In *Proc. Interspeech 2017*, pages 2655–2659.