Speaker Diarization as Sequence Generation with Speech Language Models

Anonymous ACL submission

Abstract

Recent advances in Speech Language Models (SpeechLMs), which integrate large language models with speech foundation models, have enabled unified sequence modeling of speech processing tasks. However, most LLM-based approaches to speaker diarization are tightly coupled with automatic speech recognition (ASR) and evaluated using word-level metrics, making it difficult to assess diarization behavior independent of ASR accuracy. In this work, we investigate SpeechLM as a standalone sequence modeling backbone for speaker diarization, formulating diarization as a sequence prediction problem conditioned on acoustic input. We systematically compare two output representations: an event-based representation that explicitly models speaker turn onset and offset timestamps, and a frame-based representation that predicts frame-level speaker activity. To provide structured conversational cues, we further incorporate auxiliary tasks including speech activity detection, overlapped speech detection, and speaker turn counting within the output sequence. Across multiple conversational datasets, we find that event-based representations yield more robust and consistent diarization behavior than frame-based alternatives, particularly for long-form recordings. Our results offer insight into how LLM-style architectures internalize speaker-related structure from acoustic signals, and achieve competitive diarization performance on multiple long-form datasets.

1 Introduction

Speaker diarization (SD), determining "who spoke when", is essential for downstream tasks, such as meeting transcription and conversational analysis. Traditional pipelines rely on multiple modules, including speech activity detection, segmentation, speaker embedding extraction, and clustering (Park et al., 2022), while end-to-end neural diarization (EEND) (Fujita et al., 2019b) and its variants (Fujita et al., 2019c; Horiguchi et al., 2022; Liu et al., 2021; Rybicka et al., 2024; Wang et al., 2023) trained on large-scale simulated mixtures and jointly learn speaker activity and identity to reduce error propagation. Recent research uses self-supervised learning (SSL) models further improve diarization performance (Han et al., 2025a). However, these systems are explicitly designed for SD and do not naturally extend to unified sequence modeling with other speech processing tasks.

Recent advances in SpeechLMs, which integrate large language models (LLMs) with speech foundation models, have enabled unified sequence modeling of acoustic and linguistic signals within a single autoregressive framework. These models have shown promise across a range of speech processing tasks, including ASR, text-to-speech, and speaker-related labeling (Arora et al., 2025; Cui et al., 2025). However, it remains unclear whether SpeechLM-style architectures can support fine-grained temporal reasoning tasks, such as SD, directly from acoustic signals, without relying on ASR-derived word boundaries or specialized diarization pipelines. Most existing joint ASR-diarization approaches (Wang et al., 2024a; Park et al., 2024; Efstathiadis et al., 2025) focus on refining or correcting speaker labels produced by ASR systems, and are typically evaluated using word-level metrics such as speaker-attributed or concatenated minimum-permutation word error rate (cp-WER). While effective for improving end-to-end transcription quality, this coupling makes it difficult to isolate diarization capability from ASR accuracy. Moreover, reliance on word boundaries introduces brittleness in spontaneous, overlapping, or low-resource speech, where ASR errors are frequent and unevenly distributed. As a result, the intrinsic diarization behavior of SpeechLMs, independent of lexical recognition, remains underexplored.

In this work, we study SpeechLM as a stan-

dalone generative backbone for SD, formulating diarization as token-based sequence prediction conditioned solely on acoustic input. We compare two output representations, `event`-based, which models speaker turn onsets and offsets, and `frame`-based, which predicts frame-level activity, and analyze how auxiliary tasks, including speech activity detection (SAD), overlapped speech detection (OD), and speaker turn counting (STC), influence autoregressive decoding. By organizing these auxiliary tasks within the output sequence, we expose the model to structured conversational cues at multiple levels of supervision. Our main contributions are as follows:

- We are among the first to provide a systematic study of SpeechLM as a standalone SD model, evaluated using diarization error rate (DER).
- We incorporate auxiliary tasks into the output sequence, providing structured intermediate cues that enhance SD performance and offer insight into how SpeechLM learns speaker-related information.
- We demonstrate strong performance across multiple long-form conversational datasets, validating the practicality of the proposed formulation.

2 Related Work

Most diarization-specialized systems rely on acoustic encoders or frame-level speaker representations and can be broadly grouped into clustering-based and EEND-based paradigms. A line of recent work explores the use of LLMs to refine speaker assignments produced by acoustic or ASR-based systems. Wang et al. (2024a) introduced DiarizationLM, which applies an LLM to correct word-level speaker labels output by ASR systems, enabling LLM-based refinement to be integrated into existing pipelines. Similarly, Park et al. (2024) proposed a contextual beam search framework that incorporates LLMs during decoding by jointly leveraging acoustic speaker probabilities and lexical context, significantly improving speaker-attributed word error rate over acoustic-only approaches. Efstathiadis et al. (2025) further demonstrated that fine-tuning LLMs to correct speaker labels from multiple ASR systems yields more robust, ASR-agnostic corrections. While effective for enhancing end-to-end transcriptions, these methods remain closely coupled with ASR outputs and are primarily evaluated using word-level metrics.

More closely related to our work, Cornell et al. (2024) proposed SLIDAR, a Whisper-style framework that jointly performs ASR and SD through token-based sequence prediction, generating word tokens together with onset–offset timestamps and speaker labels. This approach shares similarities with our `event`-based representation in its use of timestamped speaker representations. However, our work differs in two key aspects: we decouple diarization from word generation and study SD as a standalone sequence modeling task, and we explicitly incorporate diarization-specific tokens and auxiliary conversational tasks within a unified SpeechLM framework. Although auxiliary tasks such as SAD and OD have been explored in EEND-based models (Takashima et al., 2021), their integration into SpeechLM-style token-based diarization has not been previously examined.

Dataset	Train			Dev			Test		
	#files	#spk	#hrs	#files	#spk	#hrs	#files	#spk	#hrs
LibriMix	107,900	2-3	443.1	6000	2-3	15.9	6000	2-3	14.5
AMI	134	3-5	79.7	18	4	9.7	16	3-4	9.1
AISHELL-4	173	3-5	89.8	18	3-5	9.3	20	5	9.2
AliMeeting	209	2-4	111.4	8	2-4	4.2	20	2-4	10.8
Simulated	40,000	2-5	3303.8	800	2-5	65.9	800	2-5	65.9

Table 1: Statistics of datasets used in training, development, and testing set.

3 Data

Table 1 summarizes the dataset statistics for the training, development, and test partitions.

Target Data We conduct most of our experiments separately on two public datasets (both released under the CC BY 4.0 license): AMI (Renals et al., 2007), a corpus of long-form English conversations, and LibriMix (Cosentino et al., 2020), a corpus of short, synthetically generated utterances. For the AMI corpus, we train and test on the recordings of IHM-Mix, which mixes all individual headset microphone signals. For LibriMix, we generate samples from the LibriSpeech (Panayotov et al., 2015) `train-clean-300` and `train-clean-100` subsets, and merge the Libri2Mix and Libri3Mix datasets into a unified corpus containing utterances of two-speaker and three-speaker mixtures. We use 16KHz sampling rate and the *max* mode. To evaluate the generalization and robustness of our model, we further train and test on two additional long-form Mandarin conversational corpora, both released under the CC BY-SA 4.0 license: AISHELL-4 (Yihui Fu, 2021) and AliMeeting (Yu et al., 2022). The AISHELL-4 corpus contains recordings with up to seven speakers; since our model supports at most five speakers, we exclude segments involv-

ing more than five speakers. For AliMeeting, we use recordings captured by far-field microphones. For AMI, AISHELL-4, and AliMeeting, we follow the training, development, and test splits described in (Han et al., 2025b).

Simulated Data We further adapt the SpeechLM-based diarization model using a large-scale simulated dataset. We follow the simulation recipe of (Fujita et al., 2019a) to generate 3309.2 hours of conversations with two to five speakers and varying overlap ratios that closely match those in real datasets. Detailed statistics of the simulated data are provided in Appendix A.

4 Methods

4.1 Model Architecture

We use the open-source ESPnet-SpeechLM (Watanabe et al., 2018; Tian et al., 2025), which formulates speech processing tasks as autoregressive sequence modeling problems using a decoder-only Transformer. Each input speech sequence $\mathbf{x}$ and output text sequence $\mathbf{y}$ is first tokenized into discrete sequences $\tilde{\mathbf{x}} = [\tilde{x}_1, \ldots, \tilde{x}_M]$ and $\tilde{\mathbf{y}} = [\tilde{y}_1, \ldots, \tilde{y}_N]$, where M and N denote the lengths of the tokenized speech and text sequences, respectively. The SpeechLM model learns multi-stream sequence. Each speech frame $\tilde{\mathbf{x}} \in \mathbb{Z}^{M\times 9}$ is represented by eight codec tokens and one SSL token, extracted using a 25 ms window and a 10 ms frame shift from XEUS (Chen et al., 2024a). Output sequences $\tilde{\mathbf{y}} \in \mathbb{Z}^{N\times 9}$ are aligned and padded to match the speech token representation $\tilde{\mathbf{x}}$. The text token $\tilde{\mathbf{y}}$ are tokenized using a compact task-specific vocabulary encoding speaker IDs, timestamps, speaker turn transitions, and auxiliary-task markers; the complete vocabulary is provided in Appendix B. The model supports up to five speakers and timestamps spanning 30-second segments. The concatenated single sequence, $\mathbf{s} = [\tilde{\mathbf{x}}, \tilde{\mathbf{y}}]$, is used for training. The model is optimized with cross-entropy loss over $\tilde{\mathbf{y}}$, and thus learning the posterior distribution $p(\tilde{\mathbf{y}}|\tilde{\mathbf{x}})$. At inference, the system predicts $\tilde{\mathbf{y}}$ conditioned on $\tilde{\mathbf{x}}$, and detokenization yields the final output sequence $\tilde{\mathbf{y}}$. Figure 1 shows the multi-stream training set up for SAD and SD. The training loss function is defined as

$$\mathcal{L} = -\sum_n \Big[\lambda_{\text{mod}}\, \mathbb{I}(y_n \in \mathcal{V}_{\text{mod}}) \log p(y_n \mid \tilde{\mathbf{x}}, \tilde{y}_{<n}) + \lambda_{\text{text}}\, \mathbb{I}(y_n \in \mathcal{V}_{\text{text}}) \log p(y_n \mid \tilde{\mathbf{x}}, \tilde{y}_{<n})\Big]$$

where n indexes autoregressive decoding steps, y_n is the ground-truth token, and $\tilde{y}_{<n}$ denotes the previously generated tokens. $\mathcal{V}_{\text{mod}}$ and $\mathcal{V}_{\text{text}}$ denote the modality-indicator and text-token vocabularies, respectively, and $\mathbb{I}(\cdot)$ is the indicator function. The term $p(y_n \mid \tilde{\mathbf{x}}, \tilde{y}_{<n})$ denotes the autoregressive token probability. We weight modality-indicator and text-token prediction by $\lambda_{\text{mod}} = 10.0$ and $\lambda_{\text{text}} = 1.0$, respectively, to encourage the model to follow the prescribed sequence structure. For example, during text generation, the model is guided to first emit the modality-switch token Tokenizer $\in \mathcal{V}_{\text{mod}}$ (Figure 1), before generating the corresponding text token sequence $\tilde{\mathbf{y}}$.

4.2 Output Representation

We represent diarization in SpeechLM using two formulations: `event`-based and `frame`-based. This section describes both representations in detail, with a full example provided in Appendix C. For training, we use the full utterances for LibriMix. We divide long-form recordings from AMI, AISHELL-4, and AliMeeting into 30-second segments with a 10-second stride. We assign speaker IDs based on the order in which speakers appear in each recording and keep these IDs consistent across all segments within the same recording.

`Event`-based Representation For `event`-based approach, each speaker turn is encoded as a five-token tuple consisting of the speaker ID, a begin-of-time marker, the starting timestamp, the ending timestamp, and an end-of-time marker, e.g.,

$$T_{SD} = \{\texttt{<spk1>}\ \ \texttt{<bot>}\ \ \texttt{<0.0>}\ \ \texttt{<2.0>}\ \ \texttt{<eot>}\}$$

T_{SD} is an ordered token sequence and shows that `<spk1>` is speaking for the first two seconds. If none of the speaker is active, we represent the output sequence as

$$T_{SIL} = \{\texttt{<sil>}\ \ \texttt{<bot>}\ \ \texttt{<0.0>}\ \ \texttt{<2.0>}\ \ \texttt{<eot>}\}$$

`Frame`-based Representation The `frame`-based representation outputs one token per 0.1 second. For example, over the first 0.5 seconds, if `<spk1>` is active during the 0.1–0.3,s interval, the corresponding representation is

$$T_{SD} = \{\texttt{<sil> <spk1> <spk1> <sil> <sil>}\}$$

For overlapping speakers, we explicitly model up to three speakers (e.g., `<overlap_spk_1_2_3>`). When more than three speakers talk simultaneously, a rare case in conversational datasets, we map the region to a single token, `<overlap_spk_4_more>`. In AMI, with a 30-second audio input, the model

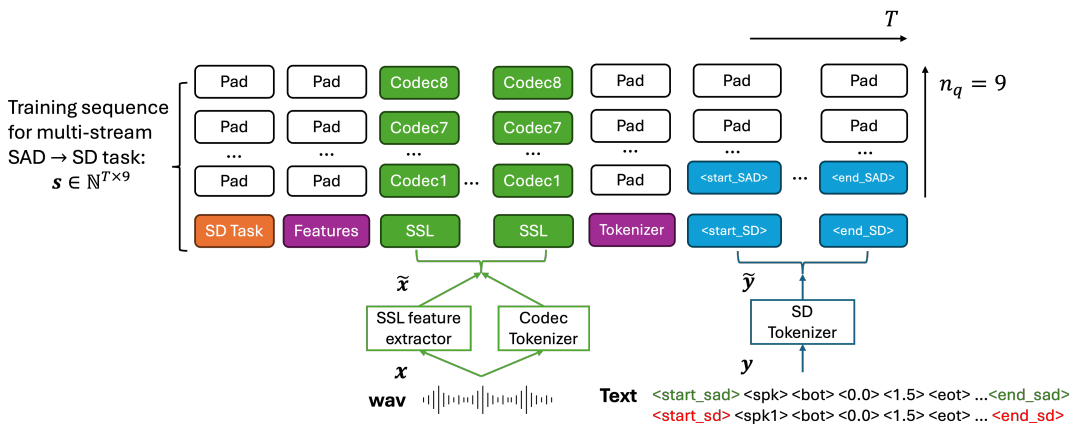


Figure 1: Overview of the SpeechLM sample training sequence in multi-stream setup. A task identifier (SD task) is followed by modality input (Features) identifier, input speech sequence, output modality identifier (Tokenizer), and output text sequence. Each speech frame contains eight codec and one SSL tokens, and the output contains SAD and SD tokens. SAD and SD are shown as separate streams. All unused tokens zero-padded.

Format	Dataset	SD	SAD	OD	STC
event	AMI	40.0 ± 21.0	18.8 ± 10.5	18.2 ± 13.3	6.7 ± 2.1
frame	AMI	323.2 ± 15.7	307.6 ± 8	307.4 ± 8.7	6.7 ± 2.1
event	LibriMix	27.9 ± 9.7	8.1 ± 3.3	14.0 ± 6.4	5.8 ± 1.0
frame	LibriMix	148.3 ± 15.9	148.3 ± 15.9	148.3 ± 15.9	5.8 ± 1.0

Table 2: Average token lengths (mean $\pm$ standard deviation) for the SD, SAD, OD, and STC tasks under `event`- and `frame`-based representations. In the `frame`-based setup, AMI includes transition markers, whereas LibriMix does not.

generates 300 frame-level tokens. For such long outputs, we add transition markers to help autoregressive SpeechLMs better align with speaker changes by indicating where turns should end. An example sequence is

T'_{SD} = {<sil> <sc_start> <spk1> <spk1> <sc_end> <sil> <sil>}

Each speaker-turn segment is augmented with explicit transition tokens, <sc_start> and <sc_end>, to mark the beginning and end of a speaker change. However, for the short synthetic conversations in LibriMix, these markers provide little benefit. Table 2 reports the average and standard deviation of token counts for each dataset under the `event`- and `frame`-based representations.

Auxiliary Tasks In addition to SD prediction, we introduce three auxiliary tasks: SAD, OD, and STC. SAD identifies speech regions within the audio, OD detects overlapping speech segments, and STC counts the number of single and overlapped speaker turns. The SAD and OD tasks can be expressed in either the `event`- or `frame`-based representation in a similar manner as SD. For STC, we instead use a single unified format:

T_{STC} = {<spk1_count> <1_count> <spk2_count> <1_count>}

Single-stream output:
Stream 2-9 output: <pad> ... <pad>
Stream 1 output: <start_sad> T_{sad} <end_sad> <start_od> T_{od} <end_od> <start_sd> T_{sd} <end_sd>

Multi-stream output:
Stream 4-9 output: <pad> ... <pad>
Stream 3 output: <start_sad> T_{sad} <end_sad>
Stream 2 output: <start_od> T_{od} <end_od>
Stream 1 output: <start_sd> T_{sd} <end_sd>

Figure 2: Sample of single-stream vs. multi-stream output for three tasks: SAD, OD and SD. Each output sequence is enclosed by task-specific markers. In the single-stream setting, the model performs stepwise sequence across tasks in the order SAD → OD → SD.

indicating that `spk1` and `spk2` each produce one speaker turn. Due to the long-tail distribution of STC, we cap the STC value at ten, grouping speakers with ten or more turns into a single `10_count` token. This token accounts for less than 2% of the corpus and avoids token sparsity.

Single-Stream and Multi-Stream Representations We explore both single-stream and multi-stream output representations for the auxiliary tasks. Figure 2 illustrates how multiple auxiliary tasks are organized under these two representations. Each task is enclosed by task-specific start and end tokens (e.g., <start_sd> and <end_sd>). In the autoregressive setting, the single-stream representation integrates all auxiliary tasks into a stepwise single sequence that guides the model toward the final SD output, while the multi-stream representation generates each task in parallel, removing this stepwise structure but reducing the overall output length. We therefore use a multi-stream setup for AMI under the `frame`-based representation, which

caps the output length at roughly 300 tokens.

For the multi-stream setup, we additionally explore assigning each speaker's activity to a separate output stream. In the `event`-based representation, each stream encodes a speaker's activity using compact onset–offset tuples (e.g., `<bot> <0.0> <2.0> <eot>`), whereas in the `frame`-based representation, each stream consists of frame-level binary activity labels (e.g., `<sil> <spk> <spk> <sil>`). This design leads to a substantial performance degradation, indicating that SpeechLM struggles to generalize when the number of output streams varies across training segments, as the number of active speakers differs across recordings. In contrast, a multi-stream decoding setup with a fixed number of output streams yields more stable performance. We therefore adopt this configuration in our study.

4.3 Post-processing

SpeechLM Decoding We use greedy decoding method at each step, with the termination symbol of task-specific marker. For example, if the final task is SD, then the decoding stops when it outputs `<end_sd>`. To extract diarization outputs, we first apply regex-based pattern matching to identify the region of SD task enclosed by task-specific marker `<start_sd>` and `<end_sd>`. For the `event`-based output, we then extract the five-token tuples and parse them accordingly. For the `frame`-based output, we retain only the tokens representing silence or speaker activity, while ignoring the transition marker tokens. If the decoded sequence exceeds the input audio duration, we discard the excess portion; if it falls short, we pad with silence. Similar decoding process has been applied to other three auxiliary tasks to examine their performance. We occasionally observe parsing failures in the `frame`-based output, as the substantially longer and marker-rich output sequences are more prone to format drift; in such cases, we perform token-level validation and retain only tokens that are valid and correctly associated with their respective tasks.

Inference For all the experiments, we do not use any oracle speaker-activity timestamps or the number of speakers during inference. For LibriMix, inference is run on each audio utterance followed by an 11-frame median filter. For AMI, AI-SHELL4, and AliMeeting, inference is performed on 30-second segments without overlap to obtain raw decoded outputs, which are then smoothed with an 11-frame median filter. To map local speaker ID across 30-second segments to global speaker ID, we follow the Pyannote inference pipeline (Bredin, 2023), as implemented in (Han et al., 2025b), to compute local speaker embeddings and align them across segments. Specifically, the raw outputs from each session are further divided into 16-second windows with a 0.1-second overlap, and speaker embeddings are extracted using a ResNet34-LM2 model trained with the WeSpeaker toolkit (Wang et al., 2024b) on VoxCeleb2 (Chung et al., 2018). We use VBx clustering (Landini et al., 2022) for post-processing, allowing 1–20 speakers per recording, with distance-based AHC initialization (threshold 0.6), foreground and background priors $F_a = 0.07$ and $F_b = 0.8$, a 128-dimensional LDA projection, and 20 VB inference iterations.

5 Experiments

Experimental Setting We fine-tune a 1.7B-parameter decoder-only Transformer with an autoregressive delay LM head (Copet et al., 2023). Hyperparameters are selected on AMI and reused for AI-SHELL4 and AliMeeting. Training uses token-based batching (batch_type=numel, where each batch contains a fixed budget of up to 45k total tokens), mixed precision, and SpecAugment (Park et al., 2019) with time masking. Optimization is performed with AdamW (Loshchilov and Hutter, 2019) (learning rate 3e-4, weight decay 1e-6), gradient clipping at 1.0, and a 4k-step warm-up, and the final model averages the three best checkpoints by development accuracy. We fine-tune the entire SpeechLM model for 10 epochs, with training time ranging from around 8 to 15 hours across datasets, depending on dataset size and representation, using a single NVIDIA H100 GPU. For domain adaptation, SpeechLM is pretrained on simulated data for 25 epochs (10k steps per epoch) using token-based batching (max 15k tokens), taking about 7 hours per epoch on a single NVIDIA A100 GPU, after which the top three checkpoints are averaged. The averaged model is fine-tuned for both dataset-specific and joint training for 3 epochs using AdamW, with learning rates of $1\mathrm{e}{-4}$ (4k-step warm-up) and $2\mathrm{e}{-4}$ (8k-step warm-up), respectively.

Evaluation Metrics We evaluate all the experiments using DER, including miss detection (MISS), false alarm (FA), and speaker confusion errors. We set zero collar tolerance and score overlapped speaker regions. For the three auxiliary tasks, we report MISS, FA, unweighted F1, and following rate (FR). SAD and OD are evaluated at a 0.1 s frame-

Setting	DER	Setting	DER
SD only	26.48	**SAD → OD → SD**	**24.83**
SD → SAD	26.22	SAD → SD → STC	25.74
SAD → SD	25.69	SAD → SD → OD	25.50
SD → OD	25.39	OD → SAD → SD	25.33
OD → SD	26.25	SAD → OD → SD → STC	25.46
SD → STC	25.79	OD → SAD → SD → STC	25.48
STC → SD	29.16	SAD → SD → OD → STC	25.68
		SAD → SD → STC → OD	26.12

Table 3: DER, in percent(%), for different training sequences incorporating various auxiliary tasks using the `event`-based representation on AMI. The best DER result is **bolded**.

Training Setting	AMI		LibriMix	
	`event`	`frame`	`event`	`frame`
SD only	26.48	37.88	13.08	12.40
SAD → SD	25.69	**29.37**	**11.64**	13.61
SD → OD	26.25	56.66	11.81	11.33
SD → STC	25.79	41.22	12.15	**10.94**
SAD → OD → SD	**24.83**	56.94	11.75	11.67
SAD→OD→SD→STC	25.46	35.96	12.08	24.52

Table 4: DER, in percent(%), for different training settings on the AMI and LibriMix datasets under `event`- and `frame`-based representations. The `frame`-based model on AMI uses a multi-stream setup, while the remaining configurations use a single-stream setup. Best DER results under each configuration are **bolded**.

level resolution, while STC is evaluated at the segment level. MISS, FA, and unweighted F1 scores measure the performance of the auxiliary-task outputs against the ground-truth labels. FR measures whether the hypothesis output derived from SD task is consistent with the auxiliary-task output, reflecting the internal consistency of SpeechLM's predictions across tasks. For example, in the STC task, FR computes the percentage where the speaker-turn count derived from the SD output matches the output directly predicted by the STC task. Formally, FR is computed over a set of evaluation units Ω. For each unit $\omega \in \Omega$, we compare the auxiliary-task prediction y_ω with the corresponding SD-derived prediction $\hat{y}_\omega$:

$$\text{FR} = \frac{1}{|\Omega|} \sum_{\omega \in \Omega} \mathbf{1}\big(y_\omega = \hat{y}_\omega\big)$$

For SAD and OD, each ω corresponds to a frame-level binary decision, so $y_\omega, \hat{y}_\omega \in \{0, 1\}$. For STC, we first apply the Hungarian algorithm to obtain a permutation-invariant alignment between the STC-predicted and SD-derived speaker counts. Each evaluation unit is then defined as a pair $\omega = (s, k)$ indexing speaker s and its k speaker turns, with $y_\omega, \hat{y}_\omega \in \mathbb{N}$ denoting the STC-predicted and SD-derived speaker turn counts for every speaker s.

6 Results

Effects of Auxiliary-Task Ordering Inspired by chain-of-thought prompting in LLM (Wei et al., 2022), we study how the ordering of auxiliary tasks, SAD, OD, and STC influences the autoregressive decoding process and the diarization performance. Table 3 reports the results under the `event`-based representation on AMI. Incorporating auxiliary tasks in a coarse-to-fine order consistently improves over the SD-only baseline (26.48% DER), which confirms the benefit of auxiliary tasks setting. Among two-stage chains, placing SAD before SD (SAD→ SD) yields better performance than the reverse order (SD→SAD) , suggesting that early exposure to speech activity cues facilitates subsequent overlap detection and speaker attribution. In contrast, placing OD or STC after SD is beneficial, as these tasks provide higher-level cues that complement diarization. Among three-stage chains, SAD→OD→SD achieves the best overall performance with 24.83% DER. Overall, these results suggest that low-level temporal cues (e.g., SAD) should precede speaker prediction, while OD is most effective when applied over speaker-active regions to learn overlap structure relevant to diarization. When auxiliary tasks are placed in this order, the model receives progressively structured intermediate signals that stabilize the autoregressive decoding process. This provides evidence that SpeechLM performs stepwise reasoning over acoustic representations rather than solely predicting diarization outputs. Adding a fourth task, enforcing STC after SD and SAD before SD, brings marginal or no further gain, suggesting diminishing returns beyond three-task chains.

`Event`- vs. `Frame`-based Representations Table 4 presents the DER across different training settings and representation formats. The results are drawn from the best configurations for each number of task chains, as identified in Table 3. For AMI, the `event`-based representation consistently outperforms the `frame`-based variant. The `frame`-based model performs reasonably well only when supported by the SAD task, but it remains underperformed by `event`-based counterpart. For LibriMix, both `event`- and `frame`-based settings perform competitively around 11–13 % DER with one-task or two-task chains. SAD task is consistently beneficial in `event`-based representation. The best

Training Settings	AMI						LibriMix					
	DER	**Task**	**Miss**	**FA**	**F1**	**FR**	**DER**	**Task**	**Miss**	**FA**	**F1**	**FR**
Event-based representation												
SD only	26.48	–	–	–	–	–	13.08	–	–	–	–	–
SAD→SD	25.69	SAD	3.3	14.1	91.3	99.9	11.64	SAD	0.8	8.5	94.9	99.9
SD→OD	25.39	OD	46.6	2.9	78.2	99.8	11.81	OD	5.3	8.3	93.2	99.9
SD→STC	25.79	STC	48.0	31.9	56.6	98.1	12.15	STC	13.4	11.1	86.6	100.0
SAD→OD→SD→STC	25.46	SAD	3.9	13.1	91.1	99.9	12.08	SAD	0.8	8.0	95.1	99.9
		OD	46.3	2.9	78.4	99.9		OD	5.3	7.5	93.7	99.9
		STC	52.1	31.1	53.5	90.3		STC	16.4	9.8	86.5	99.9
Frame-based representation												
SD only	37.90	–	–	–	–	–	12.40	–	–	–	–	–
SAD→SD	29.37	SAD	3.0	20.9	89.3	99.9	13.61	SAD	0.8	8.5	94.9	99.9
SD→OD	56.66	OD	53.7	23.7	55.8	99.9	11.33	OD	4.0	10.1	93.2	99.8
SD→STC	41.22	STC	58.2	13.5	53.8	10.4	10.94	STC	14.4	8.2	88.4	96.0
SAD→OD→SD→STC	35.96	SAD	2.9	23.9	88.3	99.9	24.52	SAD	0.4	26.5	90.4	99.9
		OD	34.4	9.5	74.0	99.9		OD	1.5	42.8	78.2	99.9
		STC	50.1	21.6	58.2	20.6		STC	15.8	5.0	89.0	98.8

Table 5: Comparison of auxiliary-task performance for AMI and LibriMix under `event`-based and `frame`-based representations. Metrics include Miss, FA, F1, and FR, with DER included for reference.

Training Method	AMI	AI-SHELL4	AliMeeting	Avg
Data-specific	24.52	13.75	29.60	22.62
+ adaptation	23.00	**12.48**	23.00	19.50
Joint	24.54	14.14	28.58	22.42
+ adaptation	**22.08**	13.29	**22.81**	**19.40**

Table 6: DER on AMI, AI-SHELL4, and AliMeeting for data-specific and joint training, with and without adaptation, using a single-stream `event`-based representation (SAD→OD→SD). The last column reports the average DER across datasets. Best DER results under each configuration are **bolded**.

performance (10.94%) arises from SD→STC under the `frame`-based setup, implying that short, synthetic data with less number of speakers benefits from frame-level modeling and explicit speaker-turn counting supervision.

Overall, the `event`-based model delivers robust performance consistently on both long-form conversational data (AMI) and short synthetic data (LibriMix). In contrast, the `frame`-based representation underperforms because these long output sequences place substantial burden on the autoregressive decoder, which must maintain fine-grained temporal consistency across hundreds of tokens. This difficulty becomes more pronounced in multi-stream decoding, where the model must also ensure cross-stream alignment, making `frame`-based modeling inherently less stable than `event`-based variant.

Analysis of Auxiliary Tasks We examine the performance of three auxiliary tasks under individual auxiliary task training (SAD→SD, SD→OD, and SD→STC) and combined task training (SAD→OD→SD→STC). Table 5 summarizes the results for AMI and LibriMix datasets. For AMI, under the `event`-based setup, the three auxiliary tasks exhibit similar performance between individual-task training and combined-task training. The SAD task provides the strongest supervision, achieving high F1 scores around 91%. The OD task offers moderate guidance with F1 scores around 78% but suffers from relatively high miss error rates (around 46%). The STC task yields lower F1 scores around 53% with higher miss and FA errors, reflecting the difficulty for SpeechLM to capture higher-level conversational structure. All three auxiliary tasks nonetheless improve DER compared to the SD-only baseline. In contrast, under the `frame`-based setup, the individual-task results show that, compared to SD-only baseline, substantially degraded OD and STC performance leads to greater DER, while reliable SAD predictions largely reduce DER. For the combined-task configuration, incorporating auxiliary tasks yields performance that slightly surpasses the SD-only baseline, though performance gain remains limited due to the instability of OD and STC supervision. For LibriMix, all three auxiliary tasks perform relatively well in both `event`- and `frame`-based representations. However, in the combined-task configuration, the `frame`-based model shows degraded SAD and OD performance, which directly contributes to significantly higher DER. Overall, the effectiveness of each auxiliary task strongly influences final diarization result: strong auxiliary-task performance provides helpful signal for SpeechLM, whereas poor auxiliary-task performance introduces noisy or con-

Model	Simulated Data Used	LibriMix	AMI	AISHELL-4	AliMeeting
Clustering-based					
Transcribe-to-Diarize (Kanda et al., 2022)	–	–	24.43	–	–
VAD+VBx+OSD (Landini et al., 2024)	–	–	22.4	15.8	28.8
pyannote 3.1 (Plaquet and Bredin, 2023)	–	–	18.8	12.2	24.4
EEND-based					
SA-EEND (Horiguchi et al., 2022)	100k mixtures on 2 speakers, ∼2433 h	–	27.7	–	–
EEND-EDA (Chen et al., 2024b)	100k mixtures on 1–5 speakers	10.16	21.56	–	–
EEND-SS (Maiti et al., 2023)	–	6.04	–	–	–
EEND-VC (Pálka et al., 2025)	–	–	14.2	9.8	12.4
Token-based					
SLIDAR (Cornell et al., 2024)	5000 h	–	27.1	–	–
SpeechLM (data-specific)	3303 h	9.09	23.00	12.48	23.0
SpeechLM (joint)	3303 h	–	22.08	13.29	22.81

Table 7: Comparison of DER (%) between SpeechLM-based diarization models with prior models in literature.

tradictory supervision that harms SD performance. Regarding FR, which evaluates the internal consistency of SpeechLM, SAD and OD maintain nearly perfect consistency across all settings and datasets. In contrast, STC shows substantially lower FR under the `frame`-based representation in AMI, which further highlighs the challenge of SpeechLM to produce consistent and robust output of higher-level speaker-turn patterns and conversational structure.

Effect of Adaption To investigate the effect of adaptation, we first train the SpeechLM-based diarization system on large-scale simulated mixtures using the best `event`-based configuration with a three-stage SAD→OD→SD chain, and then adapt the model to target datasets. We first evaluate adaptation on LibriMix, and adaption drops DER from 11.75% to 9.09%. We then study adaptation on real-world meeting datasets. For AMI, AI-SHELL4, and AliMeeting, we compare data-specific training and joint training across all three datasets, both before and after adaptation. As shown in Table 6, adaptation consistently improves DER across all datasets and training strategies. In particular, AliMeeting benefits the most from adaptation, with DER reduced from 28.58% to 22.81% under joint training. Data-specific adaptation yields larger gains on AISHELL-4, while joint training with adaptation achieves the best overall average DER, indicating robust cross-dataset generalization without dataset-specific adaptation.

Comparison with Previous Models Table 7 compares our SpeechLM-based system with representative prior approaches, grouped by modeling paradigm. Most clustering-based and EEND-based systems are explicitly designed for diarization using acoustic-only encoders or frame-level speaker representations, often with task-specific architectures and training objectives. As a result, comparisons across categories should be interpreted as indicative of relative trends rather than strictly apples-to-apples evaluations. Within the class of token-based methods, SpeechLM achieves lower DER than SLIDAR (Cornell et al., 2024) on AMI while using a smaller amount of simulated training data, and additionally supporting evaluation across multiple real-world datasets. To provide additional context beyond task-specific diarization models, we probe Google Gemini API (Anil et al., 2023) in a fully zero-shot, prompt-only setting on two AMI recordings. While the resulting DER, 76.9%, are far from competitive, highlighting the limited speaker modeling capabilities of general-purpose speech LLMs without explicit task supervision. Experimental details are included in Appendix D. Although SpeechLM is not intended to outperform specialized clustering-based or EEND-based systems in absolute DER, our results demonstrate that a flexible, token-based diarization formulation can generalize across datasets within a single unified model.

7 Conclusion

This work investigates the use of SpeechLM for SD and shows that modeling diarization as token-based sequence generation with `event`-based representations and auxiliary tasks can yield competitive performance across diverse datasets. By integrating SAD, OD, and STC cues within a single model, it consistently outperforms `frame`-based representations. Moreover, these auxiliary cues provide structured guidance that helps the model develop internal reasoning steps, and their ordering further influences SD performance. Taken together, these findings suggest that representation choice and supervision structure play a critical role in how SpeechLM-style autoregressive models capture fine-grained temporal structure from audio.

8 Limitations

Our study fine-tunes SpeechLM specifically for the SD task, which may reduce its performance on other speech processing tasks originally supported by the pretrained model. While this task-specific adaptation improves SD performance, it highlights a trade-off between specialization and maintaining general-purpose capabilities. Nevertheless, the proposed token-based sequence formulation provides a flexible modeling framework that naturally extends beyond SD. As an initial study of SpeechLM-based SD, we focus on recordings with no more than five speakers; extending the approach to denser multi-party interactions may require additional modeling or training strategies. We evaluate a single SpeechLM variant and therefore do not assess generalization across different SSL or LLM backbones, model scales, or training configurations. Our analysis of scaling effects is limited, and we do not systematically study the impact of model size, training data composition, or hyperparameter choices. Comparisons with prior diarization systems are not strictly capacity-matched. Many baseline methods are explicitly optimized for diarization, whereas SpeechLM is a general-purpose multimodal model designed to support unified modeling across multiple speech tasks. More controlled comparisons that align model capacity, training data, and computational cost, as well as extensions to joint ASR–diarization and other speech-processing tasks, remain important directions for future work.

References

Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, and 1 others. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Siddhant Arora, Kai-Wei Chang, Chung-Ming Chien, Yifan Peng, Haibin Wu, Yossi Adi, Emmanuel Dupoux, Hung-Yi Lee, Karen Livescu, and Shinji Watanabe. 2025. On the landscape of spoken language models: A comprehensive survey. *arXiv preprint arXiv:2504.08528*.

H. Bredin. 2023. pyannote. audio 2.1 speaker diarization pipeline: principle, benchmark, and recipe. In *Proc. Interspeech*, pages 1983–1987.

William Chen, Wangyou Zhang, Yifan Peng, Xinjian Li, Jinchuan Tian, Jiatong Shi, Xuankai Chang, Soumi Maiti, Karen Livescu, and Shinji Watanabe. 2024a. Towards robust speech representation learning for thousands of languages. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 10205–10224.

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

J. S. Chung, A. Nagrani, and A. Zisserman. 2018. Voxceleb2: Deep speaker recognition. In *Proc. Interspeech*, pages 1086–1090.

Jade Copet, Felix Kreuk, Itai Gat, Tal Remez, David Kant, Gabriel Synnaeve, Yossi Adi, and Alexandre Défossez. 2023. Simple and controllable music generation. *Advances in Neural Information Processing Systems*, 36:47704–47720.

Samuele Cornell, Jee-weon Jung, Shinji Watanabe, and Stefano Squartini. 2024. One model to rule them all? towards end-to-end joint speaker diarization and speech recognition. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 11856–11860. IEEE.

Joris Cosentino, Manuel Pariente, Samuele Cornell, Antoine Deleforge, and Emmanuel Vincent. 2020. LibriMix: An open-source dataset for generalizable speech separation. *Preprint*, arXiv:2005.11262.

Wenqian Cui, Dianzhi Yu, Xiaoqi Jiao, Ziqiao Meng, Guangyan Zhang, Qichao Wang, Steven Y. Guo, and Irwin King. 2025. Recent advances in speech language models: A survey. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13943–13970, Vienna, Austria. Association for Computational Linguistics.

George R Doddington, Mark A Przybocki, Alvin F Martin, and Douglas A Reynolds. 2000. The NIST speaker recognition evaluation–overview, methodology, systems, results, perspective. *Speech communication*, 31(2-3):225–254.

Georgios Efstathiadis, Vijay Yadav, and Anzar Abbas. 2025. LLM-based speaker diarization correction: A generalizable approach. *Speech Communication*, 170:103224.

Y. Fujita, N. Kanda, S. Horiguchi, Y. Xue, K. Nagamatsu, and S. Watanabe. 2019a. End-to-end neural speaker diarization with self-attention. In *Proc. ASRU*, pages 296–303. IEEE.

Yusuke Fujita, Naoyuki Kanda, Shota Horiguchi, Kenji Nagamatsu, and Shinji Watanabe. 2019b. End-to-end neural speaker diarization with permutation-free objectives. In *Interspeech 2019*, pages 4300–4304.

Yusuke Fujita, Naoyuki Kanda, Shota Horiguchi, Yawen Xue, Kenji Nagamatsu, and Shinji Watanabe. 2019c. End-to-end neural speaker diarization with self-attention. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 296–303. IEEE.

David Graff, Alexandra Canavan, and George Zipperlen. 1998. Switchboard-2 phase i. LDC98S75.

David Graff, David Miller, and Kevin Walker. 2002. Switchboard-2 phase iii audio. LDC2002S06.

David Graff, Kevin Walker, and Alexandra Canavan. 1999. Switchboard-2 phase ii. LDC99S79.

David Graff, Kevin Walker, and David Miller. 2001. Switchboard cellular part 1 audio. LDC2001S13.

David Graff, Kevin Walker, and David Miller. 2004. Switchboard cellular part 2 audio. LDC2004S07.

Jiangyu Han, Federico Landini, Johan Rohdin, Anna Silnova, Mireia Diez, and Lukáš Burget. 2025a. Leveraging self-supervised learning for speaker diarization. In *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.

Jiangyu Han, Federico Landini, Johan Rohdin, Anna Silnova, Mireia Diez, and Lukáš Burget. 2025b. Leveraging self-supervised learning for speaker diarization. In *ICASSP 2025 - 2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5.

Shota Horiguchi, Yusuke Fujita, Shinji Watanabe, Yawen Xue, and Paola Garcia. 2022. Encoder-decoder based attractors for end-to-end neural diarization. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:1493–1507.

Naoyuki Kanda, Xiong Xiao, Yashesh Gaur, Xiaofei Wang, Zhong Meng, Zhuo Chen, and Takuya Yoshioka. 2022. Transcribe-to-diarize: Neural speaker diarization for unlimited number of speakers using end-to-end speaker-attributed asr. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8082–8086. IEEE.

Federico Landini, Mireia Diez, Themos Stafylakis, and Lukáš Burget. 2024. Diaper: End-to-end neural diarization with perceiver-based attractors. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 32:3450–3465.

Federico Landini, Ján Profant, Mireia Diez, and Lukáš Burget. 2022. Bayesian hmm clustering of x-vector sequences (VBx) in speaker diarization: theory, implementation and analysis on standard tasks. *Computer Speech & Language*, 71:101254.

Yi Chieh Liu, Eunjung Han, Chul Lee, and Andreas Stolcke. 2021. End-to-end neural diarization: From transformer to conformer. In *Interspeech 2021*, pages 3081–3085.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Soumi Maiti, Yushi Ueda, Shinji Watanabe, Chunlei Zhang, Meng Yu, Shi-Xiong Zhang, and Yong Xu. 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. IEEE.

Petr Pálka, Jiangyu Han, Marc Delcroix, Naohiro Tawara, and Lukáš Burget. 2025. Vbx for end-to-end neural and clustering-based diarization. *arXiv preprint arXiv:2510.19572*.

V. Panayotov, G. Chen, D. Povey, and S. Khudanpur. 2015. Librispeech: An ASR corpus based on public domain audio books. In *Proc. ICASSP*, pages 5206–5210. IEEE.

Daniel S. Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D. Cubuk, and Quoc V. Le. 2019. SpecAugment: A simple data augmentation method for automatic speech recognition. In *Interspeech 2019*, pages 2613–2617.

Tae Jin Park, Kunal Dhawan, Nithin Koluguri, and Jagadeesh Balam. 2024. Enhancing speaker diarization with large language models: A contextual beam search approach. In *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 10861–10865.

Tae Jin Park, Naoyuki Kanda, Dimitrios Dimitriadis, Kyu J Han, Shinji Watanabe, and Shrikanth Narayanan. 2022. A review of speaker diarization: Recent advances with deep learning. *Computer Speech & Language*, 72:101317.

Alexis Plaquet and Hervé Bredin. 2023. Powerset multi-class cross entropy loss for neural speaker diarization. In *Proc. INTERSPEECH 2023*.

Steve Renals, Thomas Hain, and Hervé Bourlard. 2007. Recognition and interpretation of meetings: The AMI and AMIDA projects. In *Proc. IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 238–247. IEEE.

Magdalena Rybicka, Jesús Villalba, Thomas Thebaud, Najim Dehak, and Konrad Kowalczyk. 2024. End-to-end neural speaker diarization with non-autoregressive attractors. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*.

Yuki Takashima, Yusuke Fujita, Shinji Watanabe, Shota Horiguchi, Paola García, and Kenji Nagamatsu. 2021. End-to-end speaker diarization conditioned on speech activity and overlap detection. In *2021 IEEE Spoken Language Technology Workshop (SLT)*, pages 849–856. IEEE.

Jinchuan Tian, Jiatong Shi, William Chen, Siddhant Arora, Yoshiki Masuyama, Takashi Maekaku, Yihan Wu, Junyi Peng, Shikhar Bharadwaj, Yiwen Zhao,

and 1 others. 2025. ESPnet-SpeechLM: An open speech language model toolkit. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (System Demonstrations)*, pages 116–124.

Jiaming Wang, Zhihao Du, and Shiliang Zhang. 2023. Told: A novel two-stage overlap-aware framework for speaker diarization. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.

Quan Wang, Yiling Huang, Guanlong Zhao, Evan Clark, Wei Xia, and Hank Liao. 2024a. DiarizationLM: Speaker Diarization Post-Processing with Large Language Models. In *Interspeech 2024*, pages 3754–3758.

S. Wang, Z. Chen, and B. Han et al. 2024b. Advancing speaker embedding learning: Wespeaker toolkit for research and production. *Speech Communication*, 162:103104.

Shinji Watanabe, Takaaki Hori, Shigeki Karita, Tomoki Hayashi, Jiro Nishitoba, Yuya Unno, Nelson Enrique Yalta Soplin, Jahn Heymann, Matthew Wiesner, Nanxin Chen, Adithya Renduchintala, and Tsubasa Ochiai. 2018. ESPnet: End-to-end speech processing toolkit. In *Proceedings of Interspeech*, pages 2207–2211.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Shubo Lv Yukai Jv Yuxiang Kong Zhuo Chen Yanxin Hu Lei Xie Jian Wu Hui Bu Xin Xu Jun Du Jingdong Chen Yihui Fu, Luyao Cheng. 2021. AISHELL-4: An open source dataset for speech enhancement, separation, recognition and speaker diarization in conference scenario. In *Interspeech*.

Fan Yu, Shiliang Zhang, Pengcheng Guo, Yihui Fu, Zhihao Du, Siqi Zheng, Weilong Huang, Lei Xie, Zheng-Hua Tan, DeLiang Wang, Yanmin Qian, Kong Aik Lee, Zhijie Yan, Bin Ma, Xin Xu, and Hui Bu. 2022. Summary on the ICASSP 2022 multi-channel multi-party meeting transcription grand challenge. In *Proc. ICASSP*. IEEE.

A Information about Simulated Dataset

We follow recipe given in (Fujita et al., 2019b) and simulate conversations using datasets from Switchboard-2 (Phase I, II, III) (Graff et al., 1998, 1999, 2002), Switchboard Cellular (Part 1, Part2) (Graff et al., 2001, 2004), and NIST Speaker Recognition Evaluation datasets (2004, 2005, 2006, 2008) (Doddington et al., 2000). Given the number of speakers and the overlap parameter β, we create 5,000 simulated mixtures each for eight different settings, as summarized in Table 8.

# of speakers	β	Overlap (%)	Avg duration (s)	Total Dur. (h)
2	3	27.6	107.0	150
2	5	19.6	142.3	198.2
3	8	25.5	210.8	292
3	15	14.9	346.0	480
4	10	29.9	260.1	360
4	15	21.6	361.5	501
5	15	28.1	371.2	518
5	25	18.2	580.8	810

Table 8: Statistics of simulated mixtures under different numbers of speakers and overlap parameter β, including overlap ratio (%), average conversation duration (s), and total duration (h).

B Complete Vocabulary Inventory for SpeechLM Training

Table 9 shows the complete inventory used in this study. Transition markers are only applicable to `frame`-based representation, and time tokens are only applicable to `event`-based representation.

C Example Output Representations for LLM Diarization

Figure 3 presents example `event`-based and `frame`-based representations across four auxiliary tasks for a 2-second speaker-activity segment.

D Exploratory Experiments with the Google Gemini API

We conduct an exploratory evaluation of the latest Google Gemini API (version "gemini-3-flash-preview") in a fully zero-shot, prompt-only setting on two AMI test recordings (EN2002a and EN2002b). We focus on Gemini because it explicitly advertises support for timestamped outputs and SD, whereas other commercial audio APIs do not clearly claim such functionality and are therefore not included in this study.

Experimental Settings We follow a prompting style similar to that suggested in the official Gemini API documentation, and the full prompt is shown in Figure 4. We retain the transcription functionality while explicitly adding speaker attribution and timestamp requirements to the prompt. To ensure reproducible outputs, we set the temperature to zero so that each run produces consistent results. To process long-form AMI recordings, we segment the audio into 5-minute chunks with a 60-second overlap. We experimented with shorter chunks (e.g.,

Task	Category	Representation Used	Tokens
SD	Task-specific marker	event, frame	<start_sd>, <end_sd>
	Silence	event, frame	<sil>
	Speaker IDs	event, frame	<spk1> – <spk5>
	Overlaps for 2 speakers	event, frame	<overlap_spk_1_2>–<overlap_spk_4_5>
	Overlaps for 3 speakers	event, frame	<overlap_spk_1_2_3> – <overlap_spk_3_4_5>
	Overlaps for 4 or more speakers	event, frame	<overlap_4_more>
	Time tokens	event	<0.0> – <30.0>
	Begin- and end-of-time tokens	event	<bot>, <eot>
	Transition marker for single speaker	frame	<sc_start>, <sc_end>
	Transition marker for 2 overlap speakers	frame	<overlap_sc_start_2>, <overlap_sc_end_2>
	Transition marker for 3 overlap speakers	frame	<overlap_sc_start_3>, <overlap_sc_end_3>
	Transition marker for 4 or more overlap speakers	frame	<overlap_sc_start>, <overlap_sc_end>
SAD	Task-specific marker	event, frame	<start_sad>, <end_sad>
	Speaker active	event, frame	<spk>
	Begin- and end-of-time tokens	event	<bot>, <eot>
	Time tokens	event	<0.0> – <30.0>
	Speaker silence	frame	<sil>
	Transition marker	frame	<spk_start> – <spk_end>
OD	Task-specific marker	event, frame	<start_od>, <end_od>
	Overlap active	event, frame	<overlap>
	Begin- and end-of-time tokens	event	<bot>, <eot>
	Time tokens	event	<0.0> – <30.0>
	No overlap	frame	<no_overlap>
	Transition marker	frame	<overlap_start> – <overlap_end>
STC	Task-specific marker	event, frame	<start_count>, <end_count>
	Speaker tokens	event, frame	<spk1_count> – <overlap_4_more_count>
	Count Tokens	event, frame	<1_count> – <10_count>

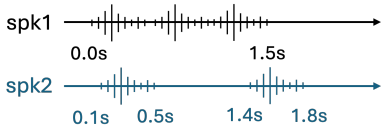
Table 9: Token inventory for SpeechLM-based diarization, organized by task, category, representation, and tokens.

file ID	DER	MISS	FA	SC
EN2002a	75.22	41.24	6.94	27.04
EN2002b	78.99	53.53	5.06	20.39
Total	76.85	46.58	6.12	24.15

Table 10: DER and its component errors (MISS, FA, and SC), in percent (%), for two AMI testing recordings.

30 seconds or 1 minute) but observed no qualitative change in the API behavior. We also tried to feed longer audio inputs, but observed that the API produced truncated outputs without returning complete transcripts. We therefore choose 5-minute chunks as a trade-off between computational cost and output stability. To mitigate hallucinated timestamps, we discard any timestamps falling outside the corresponding 5-minute chunk. To stitch adjacent segments and resolve speaker identity inconsistencies across chunks, we use the 60-second overlap to match semantically similar utterances based on their transcripts and assign global speaker labels. Finally, we merge all timestamped outputs while discarding the 60-second overlapped regions at the beginning of each 5-minute chunk.

Results Table 10 summarizes the resulting DER for the two AMI recordings. The results indicate that Gemini incurs a large proportion of missed speech segments, along with a moderate level of speaker confusion. We further inspect the generated outputs, with representative examples shown in Figure 5. One contributing factor is that Gemini fails to capture overlapping speech: all predicted timestamps are strictly monotonically increasing. Another observation is that when the temperature is set to zero, the decimal precision of timestamps shows little variation, typically ending in only one or two values (e.g., ".2" for starting timestamps or ".1" for ending timestamps in our example in Figure 5). Increasing the temperature introduces greater variability in timestamp generation, but leads to inconsistent outputs across runs and thus compromises reproducibility. Speaker confusion errors may potentially arise from the stitching procedure used to align speakers across chunks, reflecting an unavoidable complexity in modeling long-form conversational audio. Overall, these results suggest that current general-purpose speech LLMs exhibit limited speaker modeling capabilities in fully zero-



Sample speaker activity:
In the first 2 seconds, Speaker 1 speaks from 0.0 – 1.5 s,
and Speaker 2 speaks from 0.1 – 0.5 s and again from 1.4 – 1.8 s.

Event-based Representation

<start_sad> <spk> <bot> <0.0> <1.8> <eot> <end_sad>

<start_od> <overlap> <bot> <0.1> <0.5> <eot> <overlap> <bot> <1.4> <1.5> <eot> <end_od>

<start_sd> <spk1> <bot> <0.0> <1.5> <eot> <spk2> <bot> <0.1> <0.5> <eot> <spk2> <bot> <1.4> <1.8> <eot> <end_sd>

<start_count> <spk1_count> <1_count> <spk2_count> <2_count> < overlap_spk_1_2 _count> <2_count> <end_count>

Frame-based Representation (with speaker turn tokens)

<start_sad> <spk_start> <spk> ... <spk> <spk_end> <sil> <sil> <end_sad>

<start_od> <no_overlap> <overlap_start> <overlap> ... <overlap> <overlap_end> <no_overlap> ... <no_overlap> <overlap_start> <overlap> ... <overlap> <overlap_end> <no_overlap> <no_overlap> <end_od>

<start_sd> <sil> <sc_start> <spk1> <sc_end> <overlap_sc_start_2> <overlap_spk_1_2> ... <overlap_spk_1_2> <overlap_sc_end_2> <sc_start> <spk1> ... <spk1> <sc_end> <overlap_sc_start_2> <overlap_spk_1_2> <overlap_sc_end_2> <sc_start> <spk2> <spk2> <spk2> <sc_end> <sil> <sil> <end_sd>

<start_count> <spk1_count> <1_count> <spk2_count> <2_count> < overlap_spk_1_2 _count> <2_count> <end_count>

Figure 3: Example of `event-` and `frame-`based representations for SpeechLM diarization for the first two seconds. Rows from top to bottom show SAD (speech activity detection), OD (overlap detection), SD (speaker diarization), and STC (speaker turn count) outputs. Each task is embraced by the beginning-of-task and end-of-task marker. In the `frame`-based representation, each token indicates speaker activity at a 0.1s resolution. For example, in the SAD task, the token <spk> appears 18 times to represent continuous speaker activity over the first 1.8s. Speaker-turn transition tokens, such as `<spk_start>`, `<spk_end>`, `<overlap_start>`, and `<overlap_end>`, mark the boundaries of speaker changes and overlapping regions.

shot diarization settings, particularly for long-form conversational audio.

E Use of AI Assistant

We use the ChatGPT-5.2 model for occasional brainstorming, simple coding, and polishing the manuscript. All text and results in this manuscript are originally produced by the authors.

```
prompt = """You are given an audio recording.

Goal: Produce a speaker-attributed transcript.

Step 1 (ASR): Transcribe the audio with approximate timestamps.
Step 2 (Speaker attribution): Assign consistent speaker labels based on the transcript
and timing cues.

Constraints:
- Do NOT invent content. Transcript text must be verbatim (as heard).
- Timestamps should be rounded to the nearest 0.1s when possible, but may be coarser
if uncertain.
- Use labels "Speaker A", "Speaker B", ... and reuse them consistently.

Output Format:
[start_time, end_time] Speaker X: utterance
Example:
[0.0, 2.3] Speaker A: Hello, how are you?
[2.4, 4.1] Speaker B: I'm good, thank you!
"""
```

Figure 4: Prompt used for zero-shot diarization with the Google Gemini API.

```
[12.1, 14.1] Speaker A: So what do we need to talk about?
[14.2, 18.1] Speaker A: Shall we just go round and everyone says what they've what they've been
doing?
[18.2, 20.1] Speaker B: Has anybody done anything?
[23.2, 25.1] Speaker A: I've so...
[25.2, 26.1] Speaker C: Yeah.
[26.2, 28.1] Speaker B: Okay, sounds like you've done something.
[28.2, 32.1] Speaker A: Okay, well I've got a browser now, which...
[32.2, 35.1] Speaker A: Whoops, already gone. Good start.
[35.2, 37.1] Speaker B: Mhm.
[37.2, 39.1] Speaker C: Oh, I dropped mine as well.
[39.2, 43.1] Speaker B: Good at this.
[43.2, 45.1] Speaker A: This wasn't a good start.
```

Figure 5: Example of timestamped speaker outputs generated by the Google Gemini API for AMI recording EN2002a.