TRINITYASR: LEARNING LONG-RANGE SPEAKER ATTRIBUTION THROUGH IDENTITY-PRESERVING OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Speaker-Attributed, Time-Stamped Transcription (SATS) requires accurate recognition, consistent speaker attribution, and temporal localization across a conversation. In autoregressive models, an incorrect historical speaker label can affect subsequent predictions. We present **TrinityASR**, supporting up to 90 minutes of audio in a single decoding run. Our central training method, **Speaker Identity-Preserving Optimization (SIPO)**, combines corrective supervised fine-tuning (SFT) with reinforcement learning (RL). During SFT, Speaker Identity Decoupling (SID) perturbs speaker-label histories and computes optimal corrective labels from prefix matchings, accounting for arbitrary speaker-ID renaming. Subsequent RL optimizes generated transcripts with a cpCER- and DER-based reward. On AISHELL-4, the model initialized with SID achieves lower final cpCER after RL than the model initialized without SID, demonstrating the value of corrective supervision for the complete training pipeline. Training combines 7,369 hours of in-house real speech with 23,766 hours of synthetic multi-speaker conversations. Across four benchmarks, TrinityASR achieves the lowest cpCER among the evaluated systems, including 15.83% on AISHELL-4.

1 INTRODUCTION

Accurate transcripts of multi-speaker conversations are foundational to a wide range of applications, from meeting assistants and call-center analytics to assistive technologies and legal discovery (Carletta et al., 2005; Janin et al., 2003; Chen et al., 2020). In these settings, who said what, and when is as important as what was said: users need speaker-attributed, time-stamped transcripts that preserve turn structure, overlaps, and long-range references across a discussion that may span tens of minutes. We refer to this task as **Speaker-Attributed, Time-Stamped Transcription (SATS)**. Despite its practical importance, SATS is typically solved today by stitching together multiple components—automatic speech recognition (ASR) (e.g., Whisper (Radford et al., 2023)) and speaker diarization (e.g., Pyannote (Bredin et al., 2020) or x-vector clustering (Snyder et al., 2018)), each trained with different objectives and latencies, and often tuned on different datasets. Such modular pipelines are brittle: errors cascade across stages, global context is hard to leverage consistently, and end users must accept trade-offs between attribution accuracy, temporal precision, and throughput (Wang et al., 2024).

Recent advances in large language models (LLMs) and multimodal large language models (MLLMs) (Chu et al., 2023; Tang et al., 2023; Zhang et al., 2023) suggest a path toward unified solutions that jointly model audio and text. However, most existing MLLMs are developed and evaluated mainly on single-speaker speech, achieving strong ASR performance but falling short of our SATS setting, which requires jointly recognizing content, attributing it to speakers, and providing time-stamped speaker turns in multi-speaker conversations.

A practical compromise between fully modular pipelines and fully end-to-end SATS is a semi-cascaded (or hybrid) scheme: a strong ASR and an acoustic diarization front-end are still used to produce candidate words and speaker traces, while an LLM-style model is introduced as a global reconciliation layer to resolve speaker permutations, repair boundary inconsistencies around turns/overlaps, and improve the readability and consistency of the final speaker-attributed transcript.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

For example, DiarizationLM(Wang et al., 2024) post-processes the independent outputs from ASR and diarization systems with an LLM to refine speaker-attributed transcriptions, limiting its ability to correct errors that require access to the original acoustic evidence.

Recent systems increasingly integrate recognition and speaker attribution within shared models. Sortformer (Park et al., 2024) and SpeakerLM (Yin et al., 2025) explore speaker-aware modeling, while SoulX-Transcriber (Dai et al., 2026) and VibeVoice-ASR (Peng et al., 2026) generate transcripts containing speaker labels and timestamps. SoulX-Transcriber supports inputs of up to 10 minutes, while VibeVoice-ASR extends full-recording SATS to 60 minutes within a 64k-token context. These advances shift the research question from whether a model can accommodate long recordings to how training makes long context useful for speaker attribution. We therefore position long-context inference as a system capability rather than a learning contribution in itself.

Our training method addresses the sensitivity of autoregressive speaker attribution to errors in the generated speaker-label history. Scheduled sampling addresses the broader training–inference history mismatch by exposing a model to its own predictions during training (Bengio et al., 2015). SID perturbs historical speaker tags and computes a corrective label for each resulting prefix. Because speaker IDs are arbitrary within a recording, the target minimizes achievable label error under one-to-one renaming. We train on these targets, then use sequence-level feedback to optimize generated transcripts.

We introduce **TrinityASR** and **Speaker Identity-Preserving Optimization (SIPO)**. Its supervised stage, **Speaker Identity Decoupling (SID)**, perturbs contiguous spans of historical speaker tags and derives optimal continuation labels through incremental maximum-weight matching. A selective token mask handles speaker tags that span multiple tokenizer tokens. The SID-trained policy then initializes RL with a cpCER- and DER-based reward. The two stages connect explicit corrective targets with sequence-level transcription and diarization feedback. On AISHELL-4, the resulting model achieves lower final cpCER than an RL baseline initialized from SFT without SID, supporting this training design through final model quality. Our contributions can be summarized as follows:

- **Speaker Identity-Preserving Optimization.** We propose SIPO, combining optimal corrective supervision under perturbed speaker histories with cpCER- and DER-guided RL. Its speaker-label target solver uses incremental prefix matching and takes at most one sparse-graph breadth-first search (BFS) per label occurrence. SID provides the supervised initialization, and its benefit is retained in the final model: RL with SID initialization achieves lower final cpCER on AISHELL-4 than RL without it.
- **A unified long-context SATS system.** TrinityASR jointly generates transcript content, speaker identifiers, and segment timestamps with a 128k-token context window and support for up to 90-minute inputs. Inference uses one full-recording decoding run rather than independently transcribed chunks followed by speaker-label stitching.
- **Multi-regime evaluation.** TrinityASR achieves the lowest reported cpCER among the evaluated systems on AISHELL-4, Podcast, Movies, and AliMeeting. We also curate Podcast and Movies as complementary evaluation sets, with release planned to support reproducible research. Our controlled evaluation design examines supervised checkpoints, final RL models, and their evaluation cpCER trajectories.

2 Model Architecture

Given an input waveform x, TrinityASR generates a sequence of segments

$$\mathcal{Y} = (s_1, \ldots, s_N), \qquad s_i = (t_i^{\text{start}}, k_i, w_i, t_i^{\text{end}}), \tag{1}$$

where k_i is a recording-local speaker identifier, w_i is the transcript, and t_i^{start} and t_i^{end} delimit the segment. Speaker identifiers denote an equivalence relation among segments, not identities shared across recordings. Overlapping intervals can represent speech from different speakers without requiring disjoint time segments.

Model components. As illustrated in Figure 1, TrinityASR uses the audio encoder from **Whisper medium** (Radford et al., 2023) to extract acoustic representations. A projection module maps these representations into the embedding space of **Qwen3-0.6B**, initialized from the

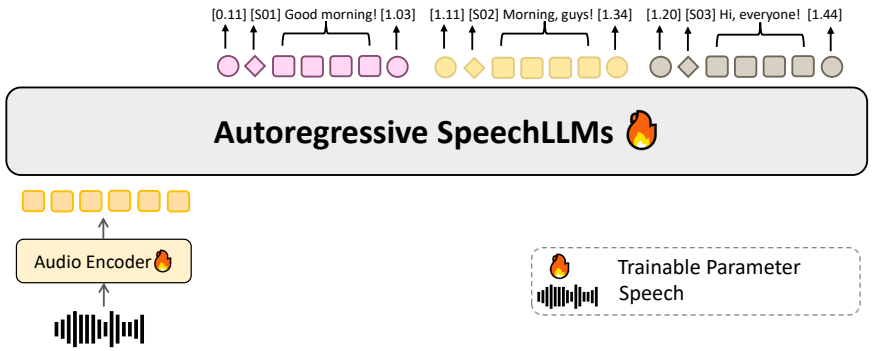

Figure 1: TrinityASR architecture: a Whisper medium audio encoder, a projection module, and a Qwen3-0.6B language model for speaker-attributed, time-stamped transcription.

`Qwen/Qwen3-0.6B` checkpoint (Yang et al., 2025; Qwen Team, 2025). Explicit temporal markers have been used to encode temporal positions in video-language models (Chen et al., 2024). In TrinityASR, we interleave time markers with the projected acoustic representations at 2-second intervals, providing temporal anchors for segment timestamp prediction. Qwen3-0.6B serves as the autoregressive language backbone and generates transcript content, recording-local speaker identifiers, and segment timestamps in the unified output format above.

Long-context decoding. TrinityASR uses a 128k-token context window and supports recordings of up to 90 minutes in a single decoding run. Here, *single-pass* refers to decoding the full recording as a single sequence without chunk-wise transcription or speaker-label stitching. Our focus is on the training strategy that enables reliable speaker attribution in this long-context setting.

3 Training Strategy

We train TrinityASR on 7,369 hours of in-house real speech and 23,766 hours of synthetic multi-speaker conversations. The following subsections describe the training data and its construction, followed by **SIPO**. SIPO trains correct continuation under perturbed histories during SFT, then optimizes the resulting policy with a cpCER- and DER-based RL reward.

3.1 Real Data

The real-data component of training comprises **7,369 hours of in-house speech recordings**. This corpus supplies natural speech supervision and is combined with the synthetic conversations described in Section 3.2. The 7,369-hour figure refers to real recordings. Podcast and Movies are separately curated evaluation sets, described in Section 4.1.1.

3.2 Synthetic Conversation Construction

Long-form conversations with accurate speaker and temporal annotations are costly to obtain. We supplement real recordings with **23,766 hours of synthetic multi-speaker conversations**, following prior work on conversation simulation (Park et al., 2023; Landini et al., 2022). We first generate a schedule specifying who speaks and when, then populate it with real speech segments and their transcripts from an in-house single-speaker corpus. Figure 2 summarizes the construction pipeline.

Extracting and fitting real conversation schedules. To capture real turn-taking, pauses, overlaps, and speaker recurrence, we sample real conversations and extract sequences $H = (h_1, \ldots, h_M)$, where $h_i = (b_i, e_i, k_i)$ specifies a segment's start time, end time, and recording-local speaker identifier. We fit Qwen3-8B (Yang et al., 2025) to model the autoregressive distribution of the serialized

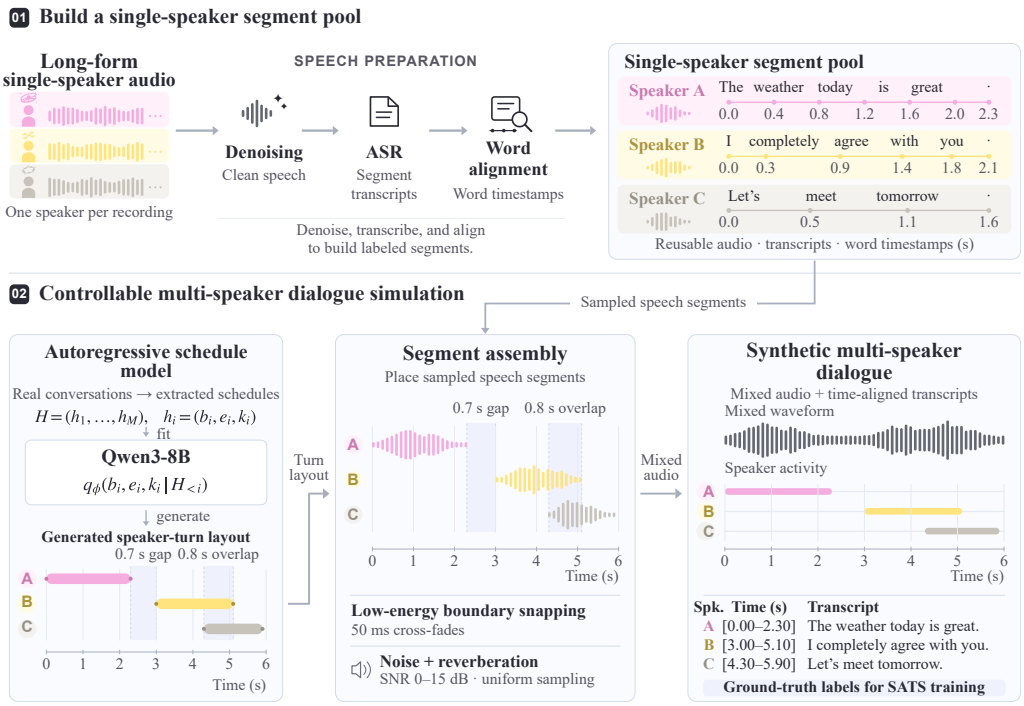

Figure 2: Synthetic conversation construction. Single-speaker recordings are denoised, transcribed, and word-aligned to form a segment pool. A Qwen3-8B schedule model fitted to real conversations supplies time intervals and speaker IDs for assembly. The resulting audio is paired with source transcripts, speaker IDs, and rendered timestamps.

triples:

$$q_\phi(H) = \prod_{i=1}^{M} q_\phi(b_i, e_i, k_i \mid H_{<i}). \tag{2}$$

At synthesis time, the model generates successive triples conditioned on the preceding schedule; it predicts neither transcript content nor waveforms.

Rendering multi-speaker audio. For each mixture, we select 2–12 source speakers and assign each generated speaker ID to one source speaker through a fixed, one-to-one mapping. We place that speaker's real speech segments on the generated timeline, ensuring that recurring IDs retain the same acoustic identity across turns. The SATS targets combine the source transcripts, mapped speaker IDs, and actual rendered timestamps. We refine segment boundaries and add noise and reverberation; Appendix A.5 provides the rendering parameters.

3.3 Speaker Identity-Preserving Optimization

Speaker Identity-Preserving Optimization (SIPO) combines two stages: supervised correction under perturbed speaker histories, followed by reinforcement learning on complete generated transcripts. Let $y^\star$ be the tokenized reference transcript for recording x, containing timestamps, speaker tags, and words. Both stages use its original speaker grouping while allowing arbitrary one-to-one renaming of speaker IDs.

3.3.1 Stage I: Speaker Identity Decoupling

Speaker Identity Decoupling (SID) perturbs contiguous windows of speaker tags to form decoder input $\tilde{y}$, preserving the audio, words, and timestamps. Reference and perturbed labels both follow

first-occurrence numbering: each new speaker receives the largest preceding ID plus one. SID constructs corrective targets for the resulting histories instead of requiring the original reference IDs.

Let $K^\star = (k_1^\star, \ldots, k_N^\star)$ and $\widetilde{K} = (\widetilde{k}_1, \ldots, \widetilde{k}_N)$ be the reference and perturbed label sequences. At occurrence i, we match labels in $\widetilde{K}_{<i}$ to those in $K^\star_{<i}$, weighting each observed pair by its number of co-occurrences. Let $h_i = \max_{j<i} \widetilde{k}_j$, with $h_1 = 0$. For any maximum-weight matching $\mathcal{M}_{i-1}$ on these observed pairs, the target is

$$k_i^\dagger = \begin{cases} u, & (u, k_i^\star) \in \mathcal{M}_{i-1}, \\ h_i + 1, & k_i^\star \text{ is unmatched.} \end{cases} \tag{3}$$

This decision minimizes the achievable number of speaker-label mismatches over valid completions of the fixed prefix, under optimal one-to-one ID renaming. Each query uses the original perturbed prefix; earlier targets are not fed into later queries. Appendix A.4 gives the formal objective, optimality proof, and incremental matching solver.

We encode these labels into an aligned target stream ℓ, retaining reference word and timestamp targets, and minimize

$$\mathcal{L}_{\mathrm{SID}}(\theta) = -\mathbb{E}_{(x,y^\star),\, \widetilde{y} \sim q_{\mathrm{corr}}(\cdot \mid y^\star)} \left[\sum_t m_t \log p_\theta(\ell_t \mid x, \widetilde{y}_{<t}) \right], \tag{4}$$

where q_{corr} is the history-perturbation distribution and m_t is a binary loss mask. Speaker tags such as `[S01]` span multiple tokenizer tokens. Within each input–target tag pair, we supervise the first differing token and mask later differences to avoid training against an inconsistent within-tag prefix; matching tokens retain their original mask. Appendix A.4 specifies the perturbations, alignment, and masking, including the `[S01]`–`[S10]` case.

3.3.2 Stage II: Metric-Guided Consistency Reinforcement

Starting from the SID-trained model, we sample groups of complete transcripts and optimize them with Group Relative Policy Optimization (GRPO) (Shao et al., 2024). For candidate $\widehat{y}_i$, the reward is

$$r_i = \max\left\{-5.0,\ 1 - 0.5\, e_i^{\mathrm{cp}} - 0.2\, e_i^{\mathrm{der}} - 0.2\, f_i\right\}, \tag{5}$$

where $e_i^{\mathrm{cp}} = \mathrm{cpCER}(\widehat{y}_i, y^\star)/100$ and $e_i^{\mathrm{der}} = \mathrm{DER}(\widehat{y}_i, y^\star)/100$ are fractional errors, and f_i indicates an invalid output format. cpCER assesses transcript content and speaker grouping; DER assesses temporal speaker assignment. This sequence-level feedback complements SID's local corrective supervision. Appendix A.1 provides reward normalization and training settings; Section 5.1 evaluates the contribution of SID initialization to final RL performance.

4 Evaluation

4.1 Evaluation Setups

4.1.1 Evaluation Datasets

We evaluate on four benchmarks. The **AISHELL-4 Test** set contains long-form meeting recordings (Fu et al., 2021); we use the average of its far-field channels for evaluation. The **Podcast** set contains multi-guest interviews from YouTube, with available platform subtitles used as the source of reference text. The **Movies** set consists of short segments from online films and TV series with frequent speaker changes and overlaps. It primarily features Chinese and English, with additional languages and dialects including Korean, Japanese, and Cantonese; its samples were manually annotated by professionals. Podcast and Movies provide reference transcripts and speaker labels but lack reference timestamps; we therefore report CER and cpCER, but not DER, on these two datasets. We also report results on **AliMeeting** from the M2MeT challenge (Yu et al., 2022) as a held-out meeting benchmark. Release of the internally curated Podcast and Movies sets is planned. Table 1 summarizes the evaluation datasets and their available duration and speaker-count statistics.

Dataset	Duration Range (s)	Avg. Duration (s)	Number of Speakers
AISHELL-4 Test	2195.4 – 2393.9	2290.6	5 – 7
Podcast	1528.7 – 4736.5	2658.9	2 – 11
Movies	0.418 – 29.888	11.526	1 – 6
AliMeeting	1572 – 2238	1890	2 – 4

Table 1: Statistical overview of the evaluation datasets.

4.1.2 METRICS

We report CER, cpCER, and DER as percentages (lower is better), and $\Delta\mathrm{cp}$ in percentage points.

CER measures character edit distance between predicted and reference transcripts, normalized by the reference character count and ignoring speaker identities (Levenshtein, 1966).

cpCER, the character-level counterpart of cpWER (Watanabe et al., 2020), concatenates transcripts by speaker within each recording and minimizes the total character edit distance over speaker permutations, normalized by the total reference character count. It evaluates recognition and speaker grouping jointly.

DER measures time-based diarization errors after optimal one-to-one speaker mapping (dscore contributors):

$$\mathrm{DER} = \frac{T_{\mathrm{miss}} + T_{\mathrm{fa}} + T_{\mathrm{conf}}}{T_{\mathrm{ref}}} \times 100\%, \tag{6}$$

where the numerator sums missed-speech, false-alarm, and speaker-confusion durations, and T_{ref} is total reference speaker time in the scored regions, counting active speakers separately when overlap is scored.

$\Delta\mathrm{cp} = \mathrm{cpCER} - \mathrm{CER}$ is a descriptive gap, not an isolated speaker-attribution error rate; it can be negative because the metrics use different transcript orderings and matching procedures.

4.2 PERFORMANCE

We compare with the systems evaluated in our study: Gemini 2.5 Pro[1], Gemini 3 Pro[2], and VibeVoice-ASR (Peng et al., 2026). The results are shown in Table 2.

Full-system results. TrinityASR achieves the lowest CER and cpCER among the available results on all four benchmarks. On AISHELL-4, cpCER is 15.83%, compared with 24.99% for the strongest reported baseline on that metric. On Podcast, it is 7.37%, compared with 10.23%. On Movies and AliMeeting, it is 12.76% and 22.17%, respectively, compared with 14.73% and 29.33%. These results support the effectiveness of the complete system for speaker-attributed transcription across the evaluated domains.

Interpreting duration and attribution. AISHELL-4 and Podcast average approximately 38 and 44 minutes, whereas Movies averages 11.5 seconds. TrinityASR performs well in both regimes, but these datasets also differ in acoustics, language, overlap, and speaker count. Their contrast does not isolate a context-length effect or establish the cause of particular errors. Similarly, the $\Delta\mathrm{cp}$ values of 0.99 on AISHELL-4 and 1.40 on Podcast are descriptive gaps, not direct measurements of speaker-memory quality. The negative value on AliMeeting (-2.69) reflects the different matching procedures used by CER and cpCER; it is not evidence that assigning speakers denoises the transcript.

System comparison versus causal evidence. VibeVoice-ASR provides a relevant long-context comparison, but differences in encoder, backbone, training data, optimization, and output conventions prevent it from serving as a controlled ablation of SIPO. The main table reports system-level performance; the SID and RL comparisons in Section 5 examine the training method within the same

[1] https://ai.google.dev/gemini-api/docs/models?hl=zh-cn#gemini-2.5-pro

[2] https://ai.google.dev/gemini-api/docs/models?hl=zh-cn#gemini-3-pro

Dataset	Metric	Gemini 2.5 Pro	Gemini 3 Pro	VibeVoice	SoulX-Transcriber	Ours
AISHELL-4	CER (↓)	42.70	22.75	21.40	\	**14.84**
	cpCER (↓)	53.42	27.43	24.99	\	**15.83**
	Δcp	10.72	4.68	3.59	\	**0.99**
	DER (↓)	15.32	22.03	**6.77**	\	9.88
Podcast	CER (↓)	7.38	\	27.94	\	**5.97**
	cpCER (↓)	10.23	\	48.30	\	**7.37**
	Δcp	2.85	\	20.36	\	**1.40**
Movies	CER (↓)	15.46	8.62	14.59	11.78	**6.36**
	cpCER (↓)	24.15	14.73	42.54	25.53	**12.76**
	Δcp	8.69	**6.11**	27.94	13.76	6.40
AliMeeting	CER (↓)	27.43	26.75	27.40	\	**24.86**
	cpCER (↓)	41.64	32.84	29.33	\	**22.17**
	Δcp	14.21	6.09	1.93	\	**-2.69**
	DER (↓)	31.60	38.75	10.92	\	**7.28**

Note: DER is reported only for AISHELL-4 and AliMeeting, which provide reference timestamps. A slash or dash denotes an unavailable result under our evaluation setup. SoulX-Transcriber results are unavailable on AISHELL-4, Podcast, and AliMeeting because the recordings exceed its supported input duration under full-recording evaluation. DER scores for Gemini 2.5 Pro, Gemini 3 Pro, and VibeVoice on AISHELL-4 and AliMeeting are taken from Peng et al. (2026). Gemini 3 Pro has results on AISHELL-4 but no valid reported result on Podcast. Δcp is descriptive; bold entries mark row minima, not an independent performance ranking.

Table 2: Performance of TrinityASR and the evaluated systems. Best CER, cpCER, and DER values and the smallest Δcp in each dataset are in **bold**.

model family. The Podcast evaluation set includes recordings up to 4,736.5 seconds (approximately 78.9 minutes), extending evaluation beyond one hour and into the 60–90-minute range (Table 1). The reported Podcast scores summarize the full evaluation set.

5 Ablation Study

Our main ablations on AISHELL-4 examine whether SID improves supervised training, whether its advantage persists after RL, and how data composition and speaker-history perturbations relate to performance. Table 3 compares four checkpoints, and Figure 3 tracks recognition and speaker-attributed transcription during RL. Section 5.2 analyzes error correction on 32 recordings from an internal two-speaker dialogue dataset. Table 5 compares training data and augmentation choices.

5.1 Final RL Performance with and without SID

We compare four checkpoints: **A**, SFT without SID; **B**, SFT with SID; **C**, RL initialized from A; and **D**, RL initialized from B (full SIPO). The main comparison is D versus C: SID initialization produces the better final model on AISHELL-4. A versus B characterizes performance before RL. Different RL starting points are part of this intervention, and both are shown explicitly in the evaluation curves.

SID improves the supervised starting point. Comparing A and B in Table 3, SID reduces cpCER from 29.39% to 22.04%, an absolute reduction of 7.35 percentage points (25.0% relative). Thus, the benefit of SID is already present before sequence-level reinforcement: the SID configuration improves speaker-attributed transcription in the reported SFT comparison. This comparison evaluates the complete SID procedure; it does not separately isolate the target solver, history perturbation, or token mask.

Condition	cpCER ↓	DER ↓
A: SFT without SID	29.39	15.76
B: SFT with SID	22.04	12.68
C: RL from A	24.30	16.72
D: full SIPO	15.83	9.88

Table 3: Supervised and final RL checkpoints, evaluated on AISHELL-4 by cpCER and DER (%; lower is better). The upper block shows the SFT models; the lower block shows their RL counterparts. C and D use the same RL budget and procedure. D versus C is the primary comparison of final model quality.

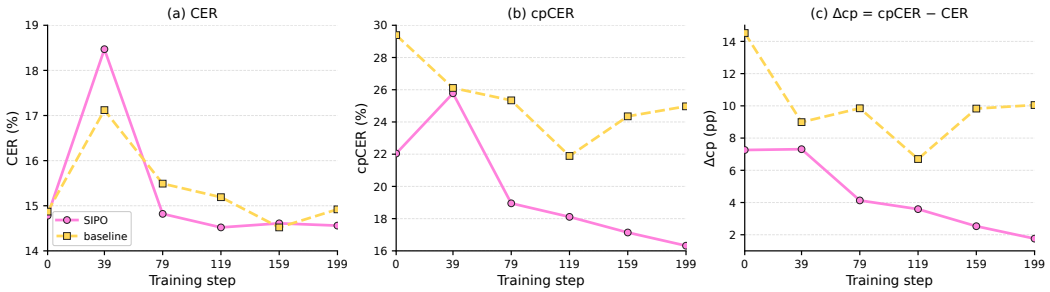

Figure 3: RL learning curves on AISHELL-4: (a) CER, (b) cpCER, and (c) $\Delta\mathrm{cp} = \mathrm{cpCER} - \mathrm{CER}$ (percentage points). SIPO uses SID-based SFT initialization; the baseline uses SFT without SID. Both trajectories include their respective SFT checkpoints at step zero. Lower CER and cpCER indicate better performance; the difference is a descriptive diagnostic.

RL further improves the SID-trained model. RL reduces cpCER from 22.04% to 15.83% along the SID path, compared with 29.39% to 24.30% without SID. The corresponding absolute reductions are 6.21 and 5.09 percentage points. The SID path therefore exhibits a larger observed reduction as well as a better final checkpoint. Because the two paths start from different SFT models, the difference between these reductions does not by itself establish that SID makes RL intrinsically more effective. The primary result is the final-model comparison: full SIPO achieves 8.47 percentage points lower cpCER than RL without SID, a relative reduction of 34.9%. SID's supervised advantage is therefore retained after RL, and RL provides an additional improvement over SID-based SFT. The optimizer and rollout settings are listed in Appendix A.1; the reported scores are point estimates rather than estimates of variation across training seeds.

Recognition and speaker attribution during RL. Figure 3 complements the checkpoint comparison with CER and cpCER trajectories, starting from the respective SFT models at step zero. At the last plotted step, SIPO has lower cpCER than the baseline despite similar CER. This pattern indicates that the cpCER advantage is not explained solely by better speaker-independent recognition and is consistent with improved speaker attribution. The smaller $\Delta\mathrm{cp} = \mathrm{cpCER} - \mathrm{CER}$ provides a descriptive view of this pattern, but is not an isolated measure of attribution error.

5.2 Error-Correction Analysis

To examine correction after attribution errors, we compare SFT checkpoints with and without SID on the same 32 recordings from an internal two-speaker dialogue dataset. Error Propagation Rate (EPR) measures how often an attribution error is immediately followed by another error, while Error Burst Rate at three (EBR@3) measures the fraction of error runs spanning at least three consecutive speaker blocks. Appendix A.6 defines the blocks, character alignment, and evaluability rules.

Table 4 shows lower EPR (30.99% versus 54.12%) and EBR@3 (2.88% versus 11.11%) for the SID-trained checkpoint, indicating less persistent attribution errors and fewer long error runs under

Checkpoint	EPR ↓	EBR@3 ↓
SFT with SID	30.99 (44/142)	2.88 (3/104)
SFT without SID	54.12 (92/170)	11.11 (10/90)

Table 4: Error-correction diagnostics on 32 recordings from an internal two-speaker dialogue dataset. Percentages are micro-aggregated, with numerator/denominator counts in parentheses. Each model uses the same evaluability rules independently; EPR counts adjacent block pairs, whereas EBR@3 counts error runs.

	AISHELL-4		
Configuration	CER ↓	cpCER ↓	Δcp
Full data	14.78	22.04	7.24
Real data only	17.17	27.42	10.25
Independent tag replacement	16.28	29.44	13.16

Table 5: AISHELL-4 controls for data composition and speaker-history perturbations. The data-composition comparison concerns the inclusion of synthetic conversations and does not isolate the synthesis method. Final RL performance with and without SID is compared separately in Table 3.

this protocol. These descriptive diagnostics complement the aggregate transcription and diarization metrics.

5.3 Data and Component Controls

Table 5 compares the full-data configuration with real-only training and independent speaker-tag replacement. We use the full-data row as the reference for these reported component comparisons; the full SIPO result after RL is presented separately in Table 3.

Contribution of synthetic conversations. The full-data configuration achieves 14.78% CER and 22.04% cpCER, compared with 17.17% and 27.42% for real-only training. The reductions of 2.39 and 5.38 percentage points, respectively, show better performance with the real–synthetic mixture on both metrics. This result supports including synthetic conversations in the training recipe, but does not establish the superiority of the learned conversation scheduler over other synthesis methods.

Choice of speaker-history perturbation. Independent tag replacement yields 29.44% cpCER, compared with 22.04% for the full-data configuration, and also has higher CER (16.28% versus 14.78%). The observed gap is consistent with the choice of perturbation being important: arbitrary tag replacement does not reproduce the performance of the SID configuration. Isolating the effect of perturbation structure, however, also requires matching the training conditions and the amount of corruption. The reported scores alone do not identify which aspect of the perturbation accounts for the gap.

6 Conclusions

We presented **TrinityASR** for long-context SATS and **Speaker Identity-Preserving Optimization (SIPO)** for training consistent speaker attribution. SIPO connects explicit corrective targets under perturbed histories with cpCER- and DER-based feedback on complete generated transcripts. On AISHELL-4, RL initialized with SID achieves lower final cpCER than RL initialized without SID, establishing the practical value of this supervised preparation for the final model. TrinityASR supports a 128k-token context window and up to 90 minutes of audio, and achieves the lowest CER and cpCER among the evaluated systems on four benchmarks.

Reproducibility Statement

Section 2 identifies the Whisper medium audio encoder and the `Qwen/Qwen3-0.6B` language backbone; Sections 3.1 and 3.2 specify 7,369 hours of in-house real training data and 23,766 hours of synthetic conversations, respectively. Table 1 provides evaluation-set statistics, Appendix A.2 contains prompts for Gemini and TrinityASR, and Appendix A.3 describes the text-normalization rules. Section 3.2 identifies the Qwen3-8B schedule generator and defines its timestamp–speaker targets and source-audio rendering. Section 3.3 defines SIPO's corrective and reinforcement stages, while Appendix A.4 specifies history perturbations, optimal corrective targets, the incremental matching algorithm and its proof, and supervision masks for non-atomic speaker tags. Appendix A.1 reports the SFT and RL optimizer settings, RL hardware and parallelism, rollout batch and sampling configuration, GRPO clipping, metric weights, the format penalty, and the reward floor. Section 5.1 defines the four training conditions and reports evaluation trajectories and checkpoint scores. Appendix A.6 defines the error-correction diagnostics and their evaluability rules. Release of Podcast and Movies is planned to facilitate reproducible evaluation.

Ethics Statement

Training uses 7,369 hours of in-house real speech and 23,766 hours of synthetically composed multi-speaker audio. The curated Podcast and Movies evaluation sets will be released for research use only, with documentation describing their sources and annotation process. Speaker-attributed transcription technology may raise privacy concerns when applied to recorded conversations; we encourage deployments to obtain appropriate consent from participants.

AI Use Statement

Generative AI tools were used to assist with manuscript drafting and editing.

References

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. *arXiv preprint arXiv:1506.03099*, 2015. URL `https://arxiv.org/abs/1506.03099`.

Hervé Bredin, Ruiqing Yin, Juan Manuel Coria, Gregory Gelly, Pavel Korshunov, Marvin Lavechin, Diego Fustes, Hadrien Titeux, Wassim Bouaziz, and Marie-Philippe Gill. Pyannote.audio: neural building blocks for speaker diarization. In *ICASSP*, pp. 7124–7128, 2020.

Jean Carletta, Simone Ashby, Sophie Bourban, Mike Flynn, Maël Guillemot, Thomas Hain, Jaroslav Kadlec, Vasilis Karaiskos, Wessel Kraaij, Melissa Kronenthal, et al. The ami meeting corpus: A pre-announcement. In *Machine Learning for Multimodal Interaction*, 2005.

Shimin Chen, Xiaohan Lan, Yitian Yuan, Zequn Jie, and Lin Ma. Timemarker: A versatile video-llm for long and short video understanding with superior temporal localization ability. *arXiv preprint arXiv:2411.18211*, 2024.

Zhuo Chen, Takuya Yoshioka, Liang Lu, Tianyan Zhou, Zhong Meng, Yi Luo, Jian Wu, Xiong Xiao, and Jinyu Li. Continuous speech separation: Dataset and analysis. In *Proceedings of ICASSP*, pp. 7284–7288, 2020. doi: 10.1109/ICASSP40776.2020.9053426.

Yunfei Chu, Zhifang Jin, Weihan Xu, Yuchen Wei, et al. Qwen-audio: Advancing universal audio understanding via unified large-scale audio-language models. *arXiv preprint arXiv:2311.07919*, 2023.

Yuhang Dai, Haopeng Lin, Zhennan Lin, Jiale Qian, Jun Wu, Hanke Xie, Hao Meng, Hanlin Wen, Chuang Ding, Shunshun Yin, Ming Tao, Lei Xie, and Xinsheng Wang. SoulX-Transcriber: A robust end-to-end framework for multi-speaker speech transcription, 2026.

dscore contributors. dscore: Diarization scoring tools. `https://github.com/nryant/dscore`. Accessed September 26, 2026.

Yihui Fu, Luyao Cheng, Shubo Lv, Yukai Jv, Yuxiang Kong, Zhuo Chen, Yanxin Hu, Lei Xie, Jian Wu, Hui Bu, Xin Xu, Jun Du, and Jingdong Chen. AISHELL-4: An open source dataset for speech enhancement, separation, recognition and speaker diarization in conference scenario. In *Proceedings of Interspeech*, pp. 3665–3669, 2021. doi: 10.21437/Interspeech.2021-1397.

Adam Janin, Don Baron, Jane Edwards, Dan Ellis, David Gelbart, Nelson Morgan, Barbara Peskin, Thilo Pfau, Elizabeth Shriberg, and Andreas Stolcke. The icsi meeting corpus. In *Proceedings of ICASSP*, 2003.

Federico Landini, Alicia Lozano-Diez, Mireia Diez, and Lukáš Burget. From simulated mixtures to simulated conversations as training data for end-to-end neural diarization. In *Interspeech 2022*, pp. 5095–5099, 2022. doi: 10.21437/Interspeech.2022-10451.

Vladimir I. Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady*, 1966.

T. Park, I. Medennikov, K. Dhawan, W. Wang, H. Huang, N. R. Koluguri, K. C. Puvvada, J. Balam, and B. Ginsburg. Sortformer: Seamless integration of speaker diarization and asr by bridging timestamps and tokens. arXiv preprint, 2024.

Tae Jin Park, He Huang, Coleman Hooper, Nithin Koluguri, Kunal Dhawan, Ante Jukic, Jagadeesh Balam, and Boris Ginsburg. Property-aware multi-speaker data simulation: A probabilistic modelling technique for synthetic data generation. In *CHiME-7 Workshop*, 2023. URL https://arxiv.org/abs/2310.12371.

Zhiliang Peng, Jianwei Yu, Yaoyao Chang, Zilong Wang, Li Dong, Yingbo Hao, Yujie Tu, Chenyu Yang, Wenhui Wang, Songchen Xu, Yutao Sun, Hangbo Bao, Weijiang Xu, Yi Zhu, Zehua Wang, Ting Song, Yan Xia, Zewen Chi, Shaohan Huang, Liang Wang, Chuang Ding, Shuai Wang, Xie Chen, and Furu Wei. VibeVoice-ASR technical report. arXiv preprint, 2026.

Qwen Team. Qwen3-0.6B model card. Hugging Face, 2025. URL https://huggingface.co/Qwen/Qwen3-0.6B.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. *ICML*, 2023.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024. URL https://arxiv.org/abs/2402.03300.

David Snyder, Daniel Garcia-Romero, Gregory Sell, Daniel Povey, and Sanjeev Khudanpur. X-vectors: Robust dnn embeddings for speaker recognition. In *Proceedings of ICASSP*, 2018.

Changli Tang, Wenyi Yu, Guangzhi Sun, Xianzhe Chen, et al. Salmonn: Towards generic hearing abilities for large language models. *arXiv preprint arXiv:2310.13289*, 2023.

Q. Wang et al. Diarizationlm: Speaker diarization post-processing with large language models. In *Proc. Interspeech*, 2024.

Shinji Watanabe, Michael Mandel, Jon Barker, Emmanuel Vincent, Ashish Arora, Xuankai Chang, Sanjeev Khudanpur, Vimal Manohar, Daniel Povey, Desh Raj, David Snyder, Aswin Shanmugam Subramanian, Jan Trmal, Bar Ben Yair, Christoph Boeddeker, Zhaoheng Ni, Yusuke Fujita, Shota Horiguchi, Naoyuki Kanda, Takuya Yoshioka, and Neville Ryant. CHiME-6 challenge: Tackling multispeaker speech recognition for unsegmented recordings. *arXiv preprint arXiv:2004.09249*, 2020. URL https://arxiv.org/abs/2004.09249.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025. URL https://arxiv.org/abs/2505.09388.

H. Yin, Y. Chen, C. Deng, L. Cheng, H. Wang, C.-H. Tan, Q. Chen, W. Wang, and X. Li. Speakerlm: End-to-end versatile speaker diarization and recognition with multimodal large language models. arXiv preprint, 2025.

Fan Yu, Shiliang Zhang, Pengcheng Fu, Shuaishuai Zhang, Yue Li, Zhiqiang Xu, Meng Li, Hui Bu, Jingqing, Bin Tan, Xiaoxia Song, Gaofeng Cheng, Jianqing Gao, Xinyan Xu, Tingyan Song, and Yi Zhang. M2MeT: The ICASSP 2022 multi-channel multi-party meeting transcription grand challenge. In *Proceedings of ICASSP*, pp. 6167–6171, 2022.

Dong Zhang, Shimin Li, Xin Zhang, Jun Zhan, Pengyu Wang, Yunfei Zhou, and Xipeng Qiu. Speechgpt: Empowering large language models with intrinsic cross-modal conversational abilities. In *EMNLP*, 2023.

A Additional Details

A.1 SFT and RL Training Configuration

Table 6 summarizes the optimizer settings shared by SFT and RL. In each stage, warmup occupies the first 1% of its learning-rate schedule, followed by cosine decay from a peak learning rate of 5×10^{-6} to zero. Table 7 reports the RL execution, policy-update, sampling, and reward settings. The rollout batch counts input prompts; the global response batch counts all 16 candidates sampled for each of the 32 prompts.

Reward and policy update. For each recording, the behavior policy samples $G = 16$ candidate transcripts. The `cp_cer_reward` routine computes Equation 5 against the original reference: cpCER and DER percentages are divided by 100, the format penalty is applied once per invalid response, and the result is clipped from below at -5.0. Both metrics account for global speaker-label permutations; DER uses the speaker-labeled intervals extracted from generated timestamps (Equation 6).

GRPO normalizes rewards within each group to obtain candidate advantages:

$$\bar{r} = \frac{1}{G}\sum_{i=1}^{G} r_i, \qquad \widehat{A}_i = \frac{r_i - \bar{r}}{\mathrm{std}(r_1, \ldots, r_G) + \epsilon}. \tag{7}$$

Here, ϵ provides numerical stabilization. The policy is initialized with $\theta_{\text{init}}^{\text{RL}} = \theta_{\text{SID}}$. We use asymmetric policy-ratio bounds $[0.8, 1.28]$, truncated importance-sampling bounds $[0, 2]$, and no KL regularization. Table 7 reports these settings together with hardware, parallelism, and rollout sampling parameters.

Parameter	**SFT**	**RL**
Optimizer	Adam	Adam
Peak learning rate	5×10^{-6}	5×10^{-6}
Learning-rate schedule	Cosine decay to 0	Cosine decay to 0
Warmup ratio	0.01	0.01
Weight decay	0.01	0.01
Adam (β_1, β_2)	$(0.9, 0.95)$	$(0.9, 0.95)$
Gradient-norm clipping	1.0	1.0

Table 6: Optimizer settings for supervised fine-tuning and reinforcement learning.

A.2 Evaluation Prompts

Gemini 2.5 Pro & Gemini 3 Pro (AISHELL-4, Podcast, Movies)

```
<audio>
请将音频中的对话内容转换为文本，语言使用音频的语言，使用[S1],[S2]...等标签标注出所有说话人。
比如：[S1]你好[S2]你好[S1]你叫什么名字？[S2]我叫小明。
只需要输出最终的结果，不需要输出其他任何内容，不需要输出换行符
```

Parameter	RL configuration
Execution and rollout	
Nodes × GPUs per node	$4 \times 8 = 32$ GPUs
Rollout tensor-parallel degree	2
Training data-parallel degree	16
Input prompts per rollout step	32
Samples per prompt / GRPO group size	16
Global response batch per rollout step	512
Policy update and sampling	
Policy optimizer	GRPO
Clipping $(\epsilon_{\text{low}}, \epsilon_{\text{high}})$	$(0.2, 0.28)$
TIS clipping bounds (lower, upper)	$(0.0, 2.0)$
KL regularization coefficient	0.0
Sampling temperature	1.0
Top-p / top-k	0.95 / 50
Maximum response length	32,768 tokens
Reward	
Scoring routine	`cp_cer_reward`
Accuracy signals	cpCER and DER
cpCER / DER coefficients	0.5 / 0.2
Error scale	Fractional (e.g., $20\% = 0.20$)
Base reward	$1 - 0.5e_i^{\text{cp}} - 0.2e_i^{\text{der}}$
Format penalty	-0.2 per invalid response
Reward floor	-5.0 after the format penalty

Table 7: RL configuration. Parallelism degrees are reported separately for rollout and training. Sampling parameters apply to RL training rollouts.

TrinityASR Prompts

AISHELL-4 & Podcast

<audio>
请将音频转写为文本，每一段需以起始时间戳和说话人编号（[S01]、[S02]、[S03]...）开头，正文为对应的语音内容，并在段末标注结束时间戳，以清晰标明该段语音范围。

Movies

<audio>
请将以下对话转录为文本，使用 [S1] [S2] 等说话人标签，对于音频中的事件，使用 [event] 标签表示。富有情感的文本用<emotion>对应文本</emotion> 表示，使用 <ovl> 标签表示音频有部分重叠，<ins></ins> 标签表示音频有插入。自动检测音频的语言，说话人标签和 <ovl> <ins> 始终用英文，event 和 emotion 跟随音频语言。

A.3 Output Normalization for Evaluation

To ensure fair comparison across systems, we apply the same text normalization to both predictions and references before computing CER/cpCER/Δcp. Given a raw string x, we perform the following steps:

- **Remove parenthetical content.** We delete any text in parentheses (and the preceding whitespaces) using `\s*\(.*?\)`.
- **Remove angle-bracket tags.** We delete any substrings matching `<.*?>` (e.g., `<emotion>...</emotion>`, `<ovl>`, `<ins>...</ins>`).
- **Remove non-speaker square-bracket annotations.** We delete any square-bracketed spans `[...]` that are not speaker IDs, using the regex `\[(?!S\d+\]).*?\]`. This

keeps only speaker tags of the form `[S1]`, `[S01]`, etc., and removes other bracketed markers such as events (e.g., `[event]`).

After normalization, each hypothesis/reference contains only speaker identifiers and plain transcript text for scoring.

A.4 SIPO Stage I: Optimal Speaker-Label Correction

This section specifies the perturbation, exact target solver, and token mask used in SID. Reference annotations remain available throughout: the perturbed sequence provides decoder history, while the reference speaker grouping defines the correction objective. The aligned loss targets are constructed from both sequences.

History perturbation and numbering. The speaker-flip routine locates response-region tag spans using temporary delimiters, samples one permutation of the existing tag encodings, and applies it to selected windows of tag occurrences. Let N be the number of response-region tags. The window-sampling defaults are $L_{\max} = 10$, $C_0 = 3$, and $\rho = 0.05$. The number of attempted flips is uniform from 0 through $C_{\max} = \max(C_0, \lfloor \rho N \rfloor)$. Each attempt samples $n \in \{1, \dots, L_{\max}\}$. If $N \le n$, the attempt is skipped; otherwise its zero-based start is sampled from $\{n, \dots, N-1\}$ and its end is $\min(N, s+n)$. Windows may overlap and compose the same permutation on the current input tags. Thus ρ scales an operation-count limit, rather than a per-tag flip probability. The target solver receives valid reference and perturbed sequences: both assign a new speaker $\max(\text{preceding IDs}) + 1$. This invariant is checked on the perturbed history, independently of the reference's numbering. Waveform, lexical content, and timestamps are unchanged.

Target definition and prefix counts. Let $K^\star = (k_1^\star, \dots, k_N^\star)$ and $\widetilde{K} = (\widetilde{k}_1, \dots, \widetilde{k}_N)$ be the reference and perturbed speaker-label sequences. For equal-length label sequences, define

$$d_{\mathrm{perm}}(C, K^\star) = \min_{F \text{ injective}} \sum_{j=1}^{N} \mathbf{1}\{F(C_j) \neq k_j^\star\}, \tag{8}$$

where F is one shared label mapping for the sequence. This loss counts mismatched speaker-label occurrences; the cpCER and DER objectives used in RL are defined separately. Let $\mathcal{V}_N$ contain length-N sequences with valid first-occurrence numbering. At occurrence i, let $h_i = \max_{j<i} \widetilde{k}_j$, with $h_1 = 0$. We choose

$$k_i^\dagger \in \arg \min_{b \in \{1, \dots, h_i+1\}} \min_{\substack{C \in \mathcal{V}_N \\ C_{<i} = \widetilde{K}_{<i},\ C_i = b}} d_{\mathrm{perm}}(C, K^\star). \tag{9}$$

The suffix of C is free. Every query fixes the original perturbed prefix $\widetilde{K}_{<i}$; previously constructed targets are not fed into later queries.

For computation, define the prefix pair counts

$$W_{uv}^{(i)} = \big|\{j < i : \widetilde{k}_j = u,\ k_j^\star = v\}\big|. \tag{10}$$

Exact correction under a fixed prefix. For any fixed prefix P of length r, every full completion has loss at least $d_{\mathrm{perm}}(P, K^\star_{\le r})$. This lower bound is attainable. Choose an optimal matching between the prefix labels and the reference labels. Map unmatched prefix labels to distinct labels absent from the full reference. For every suffix position, reuse the preimage of its reference label if that preimage exists; otherwise create the next unused input ID and map it to that reference label. This preserves injectivity and first-occurrence numbering, and adds no suffix errors. Consequently,

$$\min_{C_{\le r} = P} d_{\mathrm{perm}}(C, K^\star) = d_{\mathrm{perm}}(P, K^\star_{\le r}). \tag{11}$$

The remaining $N - r$ labels can be chosen freely subject to valid first-occurrence numbering. The labels' numerical values matter only for canonical serialization; all labels absent from the prefix are equivalent under one-to-one renaming. We therefore represent the new-label choice by $h_i + 1$.

For the prefix count matrix in Equation 10, a label mapping agrees with the reference at exactly the sum of the corresponding counts. Maximizing that sum is a maximum-weight bipartite matching problem; unmatched labels may map outside the reference alphabet. If the optimal prefix score is S_{i-1}, any current target has full-completion loss at least $(i-1) - S_{i-1}$. The target in Equation 3 achieves this bound. Any maximum-weight matching suffices, including when the optimum is tied. The solver always advances its counts using $(\widetilde{k}_i, k_i^\star)$, not $(k_i^\dagger, k_i^\star)$.

Sparse matching network. Let L and R be the distinct perturbed and reference labels, represented as disjoint vertex sets, and add a hub h. For each observed pair (u, v), add a real edge $u \to v$ of capacity 1 and cost $-W_{uv}$. Add auxiliary edges $u \to h$ for $u \in L$ and $h \to v$ for $v \in R$, each of capacity 2 and cost 0. Left vertices supply one unit each, right vertices demand one unit each, and h supplies $|R| - |L|$. Each feasible integral flow selects a matching on real edges; unmatched vertices use auxiliary edges. Conversely every matching extends to such a flow. Its cost is the negative matching weight.

Initialize all label vertices and auxiliary edges before processing the sequence. Initially, every auxiliary edge carries one unit, there are no real edges, and all vertex potentials are zero. Auxiliary flows can only be zero or one because each left vertex has total outflow one and each right vertex has total inflow one. Capacity two therefore ensures that auxiliary forward residual edges always exist. The graph uses $O(M + |L| + |R|) = O(M)$ space, where M is the number of distinct observed label pairs.

One BFS per unit-weight update. Maintain integer potentials p with nonnegative reduced cost on every residual edge:

$$r(u, v) = c(u, v) + p(u) - p(v) \geq 0. \tag{12}$$

This certifies minimum flow cost and hence maximum matching weight: potentials cancel around cycles, so no residual cycle can decrease the cost. An update $W_{uv} \leftarrow W_{uv} + 1$ decreases the real forward cost by one and increases its reverse cost by one.

If the real edge is matched, only its reverse has residual capacity and its cost increases; the certificate remains valid. If it is unmatched and its new reduced cost is nonnegative, no adjustment is required. Otherwise $u \to v$ is the unique negative residual edge and has reduced cost exactly -1. Run BFS from v along zero-reduced-cost residual edges.

If u is reached, the path $v \rightsquigarrow u$ together with $u \to v$ forms a cost--1 cycle. Send one unit around it. The negative edge becomes saturated and its reverse has reduced cost $+1$; reversing the path's zero-cost edges leaves zero-cost residual edges. All residual reduced costs are again nonnegative, so this single adjustment restores optimality.

If u is not reached, let S be the full set reached by BFS and set $p(z) \leftarrow p(z) - 1$ for $z \in S$. Every residual edge leaving S previously had integer reduced cost at least one, since no zero-cost edge leaves this reachable set. Its new reduced cost is therefore nonnegative. Edges entering S increase by one, including $u \to v$, whose reduced cost becomes zero. Edges internal to either side are unchanged. This also restores the certificate without changing the matching.

Insertion of a previously unseen pair. Insert its real edge with cost zero and flow zero before applying the unit decrement. The always-present auxiliary forward residual edges imply $p(u) \geq p(h) \geq p(v)$, so the inserted edge initially has nonnegative reduced cost. The preceding update proof applies unchanged. No dense collection of zero-weight real edges is needed.

Reading a target uses the maintained right-vertex mate in constant time. Each update takes at most one $O(M)$ BFS and an $O(M)$ potential or cycle adjustment. This yields $O(NM)$ time and $O(M)$ auxiliary solver space under the usual unit-cost label-indexing model. Since $M \leq N$, the worst-case time is $O(N^2)$. For T tokenizer tokens, the token streams and masks require $O(T)$ storage and processing in addition to the label solver.

Non-atomic tags and causal masking. The mathematical matching treats a speaker ID as one label, but the tokenizer represents bracketed tags such as `[S01]` with multiple tokens. The aligned implementation requires equal token lengths for the tag encodings used in an example, including corrective targets. Temporary delimiters identify corresponding spans. Initialize ℓ from the reference

Algorithm 1 SID: optimal label targets under fixed perturbed histories

Require: reference tokens $y^\star$ and aligned perturbed input $\widetilde{y}$, with valid first-occurrence speaker numbering

extract $K^\star$, $\widetilde{K}$ and their aligned tag spans
$\ell \leftarrow \textsc{Copy}(y^\star)$; initialize valid-response loss mask m
initialize the sparse flow network, zero counts, zero potentials, and empty matching $\mathcal{M}$
$h_{\max} \leftarrow 0$
for $i = 1, \ldots, N$ **do**
    if $k_i^\star$ has mate u in $\mathcal{M}$ **then**
        $k_i^\dagger \leftarrow u$
    else
        $k_i^\dagger \leftarrow h_{\max} + 1$
    end if
    write $\textsc{EncodeTag}(k_i^\dagger)$ to tag span i in ℓ
    increment $W_{\widetilde{k}_i, k_i^\star}$; repair flow and potentials with at most one BFS
    $h_{\max} \leftarrow \max(h_{\max}, \widetilde{k}_i)$
end for
for each aligned response-tag span **do**
    $D \leftarrow$ ordered tokenizer positions where $\widetilde{y}$ and ℓ differ
    set $m_d \leftarrow 0$ for every $d \in D$ except its first element
end for
strip temporary delimiters from $(\widetilde{y}, \ell, m)$ with one shared position-selection mask
return $(\widetilde{y}, \ell, m)$ for masked causal next-token training

and replace each speaker-tag span with the encoding of its independently computed $k_i^\dagger$. Compare each target tag with its actual input tag. If $d_1 < \cdots < d_q$ are the tokenizer positions where they differ, retain $m_{d_1} = 1$ and set $m_{d_j} = 0$ for $j \geq 2$; matching tokens keep their original masks. After the first differing input token enters the causal context, supervising later differences would combine a corrupted within-tag prefix with a reference-derived suffix. A single differing token causes no additional masking. Thus the `[S01]` versus `[S10]` case is handled using tokenizer positions without assuming a character-by-character tokenization. Prompt, padding, and other excluded positions remain masked. Remove the temporary delimiters from input, targets, and mask using the same position-selection mask; the language-model loss performs the causal next-token shift.

The collection of targets need not itself be a globally renamed reference transcript. For example, with $K^\star = (1, 2, 3)$ and $\widetilde{K} = (1, 1, 1)$, the independent optimal targets are $(1, 2, 2)$: the third query again has the original prefix $(1, 1)$, so its new-label choice is 2. The targets supervise different fixed histories. The label-level theorem establishes the optimality of each complete-label decision; selective masking specifies how these decisions are used in token-level training.

Independent replacement control. The optional replacement routine supplies the isolated-error control in Section 5. It preserves the first occurrence of each tag and considers subsequent occurrences. It samples a per-example replacement probability uniformly from $[0, p_{\max}]$, with default $p_{\max} = 0.3$, and replaces selected input tags with a different existing tag. The perturbation leaves the reference annotations unchanged. It replaces complete equal-length tag encodings, rather than arbitrary digits.

A.5 Synthetic Audio Rendering Details

The acoustic renderer partitions source utterances into contiguous word runs, snaps cuts to nearby low-energy points, and uses 50 ms cross-fades to reduce discontinuities. Overlap is capped at 80% of the shorter segment. Following prior work (Landini et al., 2022), we add real-world noise and reverberation, with SNRs sampled uniformly from 0–15 dB. These operations render the generated timestamp–speaker schedules into audio; timing targets reflect the actual composition timestamps, while transcript and word-boundary accuracy depends on the source annotations.

A.6 Error-Correction Evaluation Protocol

Section 5.2 compares complete decodes of the same 32 recordings from an internal two-speaker dialogue dataset, using SFT checkpoints with and without SID.

Reference blocks and attribution. Merge adjacent same-speaker reference segments into maximal *speaker occurrence blocks*. Character-level Levenshtein alignment matches each recording's normalized reference and predicted texts. Matched and substituted characters vote for their predicted speaker IDs; each block takes the unique plurality winner. Fix one global cpCER-optimal speaker mapping per recording. Coverage is the fraction of a block's reference characters that vote. Blocks with coverage below 50%, empty text, no votes, or tied winners are NA. Apply these rules independently to each model; evaluable positions may differ.

For each recording $r = 1, \ldots, R$ ($R = 32$) with T_r reference blocks, let $c_{r,i}$ be 1 for correct attribution, 0 for incorrect attribution, and $\perp$ for NA after mapping.

Error Propagation Rate (EPR). For immediately adjacent evaluable blocks, define

$$N_{ab} = \sum_{r=1}^{R} \sum_{i=1}^{T_r-1} \mathbf{1}\{c_{r,i} = a \wedge c_{r,i+1} = b\}, \quad a, b \in \{0, 1\},$$
$$\mathrm{EPR} = \frac{N_{00}}{N_{00} + N_{01}}. \tag{13}$$

Pairs containing NA are excluded; we do not skip NA blocks to connect their neighbors or count transitions across recordings. A recording's final block has no successor and does not enter the denominator.

Error Burst Rate at three (EBR@3). Let M_r be the number of maximal consecutive zero-valued runs in recording r, and let $L_{r,m}$ be the observed length of run m. Then

$$\mathrm{EBR@3} = \frac{\sum_{r=1}^{R} \sum_{m=1}^{M_r} \mathbf{1}\{L_{r,m} \geq 3\}}{\sum_{r=1}^{R} M_r}. \tag{14}$$

Correct and NA blocks break runs; recording-end runs retain their observed lengths. We micro-aggregate counts across recordings, multiply ratios by 100, and report N/A for zero denominators. These diagnostics describe error persistence without establishing a causal propagation effect.