Beyond Transcripts: A Renewed Perspective on Audio Chaptering

Fabian Retkowski[1] **Maike Züfle**[1] **Thai Binh Nguyen**[1]
Jan Niehues[1] **Alexander Waibel**[1,2]
[1]Karlsruhe Institute of Technology [2]Carnegie Mellon University
{fabian.retkowski, maike.zuefle}@kit.edu

Abstract

Audio chaptering, the task of segmenting long-form audio into coherent sections, is increasingly important for navigating podcasts, lectures, and videos. Despite its relevance, research remains limited and text-based, leaving key questions unresolved about leveraging audio information, handling ASR errors, and transcript-free evaluation. We address these gaps through three contributions: (1) a systematic comparison between text-based models with acoustic features, a novel audio-only architecture (AudioSeg) operating on learned audio representations, and multimodal LLMs; (2) empirical analysis of factors affecting performance, including transcript quality, acoustic features, duration, and speaker composition; and (3) formalized evaluation protocols contrasting transcript-dependent text-space protocols with transcript-invariant time-space protocols. Our experiments on YTSeg reveal that AudioSeg substantially outperforms text-based approaches, pauses provide the largest acoustic gains, and MLLMs remain limited by context length and weak instruction following, yet MLLMs are promising on shorter audio.[1]

1 Introduction

As long-form audio and video content becomes increasingly common, such as podcasts, lectures and YouTube videos, users need better tools to navigate and locate information within recordings. In practice, people rarely consume these recordings linearly (Yürüm et al., 2022; Liao and Wu, 2023). Instead, they skim, scrub the timeline, jump to relevant moments, and return to specific sections. This non-linear behavior makes chapter markers a key interface for browsing and re-finding content. The task of *audio chaptering* addresses this by automatically segmenting audio into coherent sections.

Despite growing relevance, research on audio chaptering remains limited and predominantly text-based: models typically operate on transcripts and inherit evaluation protocols from text segmentation (Retkowski and Waibel, 2024; Freisinger et al., 2025). This transcript-centric framing leaves several key limitations unresolved. First, the role of audio remains unclear: because most prior work treats chaptering as a purely textual problem, there is limited understanding of how audio can be leveraged for chaptering, whether through hand-crafted acoustic features or learned representations, and whether it improves performance. Second, text-segmentation evaluation protocols assume a fixed transcript. In practice, chaptering systems often rely on ASR outputs whose errors and segmentation differences change the underlying unit sequence, especially the sentence boundaries and the number of sentences. This changes the granularity (and difficulty) of the segmentation task, so standard text-based metrics computed on different transcripts are not directly comparable and can appear to improve simply because one transcript is coarser, rather than because the model segments better. Finally, chapter boundaries are intrinsically defined in continuous time, but many pipelines "snap" these timestamps to sentence boundaries. This realignment is inherently lossy: it can shift boundaries away from their true temporal positions and can systematically bias evaluation toward sentence segmentation artifacts.

This work aims to establish a methodological foundation for audio chaptering, addressing aforementioned gaps through three main contributions:

1. We systematically evaluate three modeling paradigms for audio chaptering: text-based models with and without acoustic feature augmentation, a novel audio-only architecture (AudioSeg) that operates directly on learned speech representations, and lastly, we explore whether multimodal large language models

[1]We release the `chunkseg` evaluation package (GitHub, PyPI) and the AudioSeg model (HF); additional annotations have been merged into the YTSeg dataset repository (HF).

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 8765–8787

(MLLMs) are capable of this task.

2. Second, we provide empirical insight into factors affecting chaptering performance. We analyze the robustness of text-based models to ASR errors, quantify the contribution of different acoustic features, and examine how audio characteristics such as duration and speaker composition influence segmentation.

3. Third, we systemize evaluation for audio chaptering: we formalize existing text-based protocols, analyze their limitations, and introduce time-based evaluation that enables fair comparison across text-based, audio-only, and multimodal models independent of the transcript.

To support further work on audio chaptering, we release an evaluation toolkit and AudioSeg.[1]

2 Evaluation Protocols

2.1 Text-Based Segmentation

Segmentation has traditionally been studied in the text space $\mathcal{X}$, where a document is modeled as a sequence of discrete units (typically sentences). The objective is to identify a boundary sequence $\mathbf{y} = (y_1, \ldots, y_{N-1})$, where $y_i = 1$ denotes a boundary between units s_i and s_{i+1}. Evaluation compares a predicted sequence $\hat{\mathbf{y}}$ against a reference $\mathbf{y}$ using segmentation metrics such as P_k (Beeferman et al., 1999) and Boundary Similarity (B; Fournier 2013), as well as classification metrics such as F1 score. We adopt this formalism for audio chaptering. However, a fundamental domain mismatch exists: chapters are defined as continuous *timestamps* in the time domain $\mathcal{T}$, whereas text segmentation assumes *discrete indices*. The protocols below differ in how they map between $\mathcal{T}$ and $\mathcal{X}$, and whether the metric operates in $\mathcal{X}$ or $\mathcal{T}$.

2.2 Text-Space Protocols

In existing work, evaluation is typically defined in text space by projecting continuous-time chapter boundaries onto transcript units, either on reference transcripts (R1) or on ASR transcripts (H1). We distinguish these two settings and then introduce two hybrid variants (H2–H3) that map ASR-based predictions back to a canonical reference transcript.

2.2.1 Evaluation on Ref. Transcripts (R1)

Let the *reference transcript* be the canonical text representation of the audio, segmented into sentences $S_{\text{ref}} = (s_1, \ldots, s_N)$. Ground-truth chapter boundaries are given as timestamps $\mathcal{T}_{\text{gold}} \subset \mathbb{R}^+$ on the audio. To evaluate in text space, we first align S_{ref} to the audio signal via forced alignment (FA) or closed caption timestamps to obtain a start time $t_{\text{start}}(s_i)$ and end time $t_{\text{end}}(s_i)$ for each sentence. We define a projection function $\phi_{\text{ref}} : \mathcal{T} \to \{1, \ldots, N-1\}$ that maps each timestamp in $\mathcal{T}_{\text{gold}}$ to the nearest sentence boundary in S_{ref}.

In protocol **R1 (Ref)**, a model predicts a boundary sequence over sentences in the reference transcript, and we compute text segmentation metrics on S_{ref}. This approach has two implications: (1) continuous-time boundaries are projected to sentence boundaries (a lossy discretization); (2) evaluation assumes access to the transcript. This corresponds to the protocols used in most prior work (Lai et al., 2016; Retkowski and Waibel, 2024).

2.2.2 Evaluation on ASR Transcripts (H1)

In realistic deployments, systems operate on an *ASR transcript* rather than the reference. We therefore define protocol **H1 (ASR)**, which mirrors R1 but replaces the reference transcript with the ASR transcript. We segment the ASR output into sentences $S_{\text{asr}} = (u_1, \ldots, u_M)$, where usually $M \neq N$, and obtain timestamps for each sentence (from the ASR decoder or via FA). Each chapter timestamp in $\mathcal{T}_{\text{gold}}$ is then mapped to the nearest ASR sentence boundary in S_{asr}, yielding a gold boundary sequence over ASR sentences. This protocol is also reflected in prior work (Lai et al., 2020; Freisinger et al., 2025).

H1 removes the need for a reference transcript and directly reflects operating conditions, but scores become dependent on the particular ASR system used: different models or decoding settings can yield different segmentations, changing metrics even when the underlying time boundaries are identical. H1 is therefore *not* transcript-invariant.

2.2.3 Alignment to Reference Text (H2/H3)

To recover a canonical evaluation space while still supporting models that operate on ASR, protocols **H2** and **H3** project ASR-based predictions back onto the reference transcript S_{ref}. Both establish a monotonic mapping from ASR sentences S_{asr} to reference sentences S_{ref}, but differ in how this mapping is derived: **H2** uses token-level alignment between ASR and reference transcripts, while **H3** uses maximal temporal overlap between ASR and reference sentences. Once the mapping is established, predicted boundaries between ASR sentences are transferred to S_{ref} and evaluated using standard text segmentation metrics. Both protocols anchor scores in a single canonical transcript, improving comparability across systems. H2 leverages lexical correspondence but is sensitive to ASR errors; H3 relies solely on timestamps, reducing dependence on word-level accuracy. See Appendix F.2 for details.

2.3 Time-Space Protocols

2.3.1 Discrete-Time Evaluation (T1)

All text-space protocols above assume evaluation over a sequence of text units. For audio chaptering, however, the primary object is the *time axis*. Protocol **T1 (Time-based, discrete)** evaluates segmentation in a discretized time space without privileging any transcript. The audio duration D is discretized into fixed-length frames (e.g., $\Delta t = 1\text{s}$), yielding $K = \lceil D/\Delta t \rceil$ chunks. Both gold and predicted boundaries are mapped to these chunks, producing binary sequences of length K. This yields a boundary sequence over time chunks, directly analogous to a boundary sequence over sentences. Crucially, text segmentation metrics such as P_k operate on sequences of binary decisions and therefore apply unchanged when we replace sentences by time chunks. T1 accommodates any model whose outputs can be expressed as time intervals: audio-only models predicting on time chunks, models that output timestamps, and even text-based models, whose sentence-level boundaries can be mapped back to time via ASR timestamps or FA. Once outputs are mapped to time chunks, the transcript no longer enters the metric, making evaluation *transcript-invariant*.

2.3.2 Continuous-Time Evaluation (T2)

Finally, protocol **T2 (Time-based, continuous)** evaluates segmentation directly in continuous time. We represent the reference and predicted segmentations by ordered sets of boundary timestamps, $\mathcal{T}_{\text{gold}} = \{t_1, \ldots, t_K\}$ and $\hat{\mathcal{T}}_{\text{pred}} = \{\hat{t}_1, \ldots, \hat{t}_L\}$. Unlike T1, text segmentation metrics are not applicable in continuous space. For time-stamped events, collar-based evaluation (e.g., F1 under a fixed temporal tolerance around each boundary) is common in sound event detection and speaker diarization (Mesaros et al., 2016; Bredin, 2017). Early work applies such tolerance-based boundary scoring to topic segmentation (Guinaudeau et al., 2010), but to our knowledge, this perspective has not been made explicit or systematized. T2 is the most direct formulation of the audio chaptering task: it requires no transcript, alignment, or discretization and produces transcript-invariant scores.

3 Approaches

We investigate three paradigms for audio chaptering: (1) text-based; operating on transcripts, optionally augmented with acoustic features; (2) audio-only; without transcription; and (3) MLLMs. For text-based approaches, we build on MiniSeg (Retkowski and Waibel, 2024), a hierarchical transformer that serves as our baseline.

3.1 Text-Based Baseline

MiniSeg (Retkowski and Waibel, 2024) is a two-stage hierarchical model for text segmentation. A trainable sentence encoder (based on MiniLM; Wang et al. 2020) produces fixed-dimensional embeddings for each sentence in the transcript. These embeddings are then processed by a RoFormer document encoder, which performs sequence labeling to predict binary boundary decisions. The model is trained with weighted binary cross-entropy to address class imbalance.

3.2 Hand-Crafted Audio Features

Category	Features
Pauses ⌛	`pause_duration`
Speaking Rate ⏱	`wpm, z_wpm`
Pitch 🎵	`mean_f0, std_f0, min_f0, max_f0, range_f0, slope_f0, voicing_ratio, zF0_mean, zF0_slope`
Loudness 🔊	`lkfs, z_loudness`
Speakers 👥	`same_as_prev, speaker_change, turn_id, pos_in_turn, turn_len, dist_prev_same, num_speakers_so_far`

Table 1: Audio features extracted for each sentence

While text-based models rely primarily on the semantic content of transcripts, we hypothesize that incorporating acoustic and conversational cues can improve segmentation performance. We therefore augment MiniSeg with hand-crafted audio features extracted from the speech signal; the corresponding features are presented in Table 1 while the extraction methodology is detailed in Appendix G.

3.2.1 Feature Fusion

We integrate features using concatenation-based fusion. For each sentence, we concatenate the sentence embedding from MiniSeg's sentence encoder with the feature vector, then apply linear projection:

$$\mathbf{h}_i = \text{Linear}([\mathbf{e}_i \,||\, \mathbf{f}_i]),$$

where $\mathbf{e}_i \in \mathbb{R}^{d_{\text{emb}}}$ is the sentence embedding, $\mathbf{f}_i \in \mathbb{R}^{d_{\text{feat}}}$ the feature vector, and $\mathbf{h}_i \in \mathbb{R}^{d_{\text{emb}}}$ the fused representation passed to the document encoder.

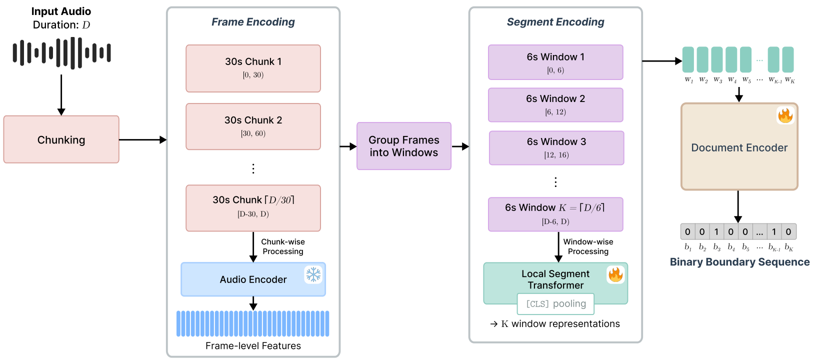

Figure 1: **AudioSeg** processes input audio of duration D through three stages: **Frame Encoding** extracts frame-level features from 30s chunks using a frozen audio encoder ❄️; **Segment Encoding** groups frames into 6s windows and encodes each via a Local Segment Transformer 🔥 with [CLS] pooling to produce $K = \lceil D/\Delta t \rceil$ segment embeddings; **Document Encoding** processes the segment sequence through a RoFormer encoder 🔥 to predict a binary boundary sequence $(b_1, \ldots, b_K)$ indicating chapter boundaries.

3.3 Audio-Only Model

We develop an audio-only segmentation model (AudioSeg) that operates directly on frame-level acoustic features without requiring a transcript. The architecture follows a three-stage processing pipeline (Figure 1): *frame encoding* transforms raw audio into frame-level representations, *segment encoding* aggregates these features into segment-level embeddings, and *document encoding* models long-range dependencies to predict chapter boundaries.

Frame Encoding. Audio is processed by a frozen, pre-trained audio encoder that produces frame-level features at a temporal stride. To handle long audio, we adopt chunked processing: the audio is divided into fixed-length chunks (e.g., 30 s), frame-level features are extracted independently for each chunk, and the resulting frames are concatenated to form a continuous sequence spanning the full audio.

Segment Encoding. The frame sequence is partitioned into non-overlapping windows of duration Δt (e.g., 6 s). Each window is encoded into a segment-level representation using a *Local Segment Transformer* which projects frames to a hidden dimension, prepends a learnable `[SEG]` token, adds positional embeddings, and applies a local transformer encoder. The resulting `[SEG]` token is taken as the segment embedding, producing $K = \lceil D/\Delta t \rceil$ segment embeddings.

Document Encoding. The segment embeddings are processed by a RoFormer encoder to model long-range dependencies across the full document. The outputs are passed through a linear layer to produce a boundary probability for each segment.

Training. Ground-truth chapter boundaries (continuous timestamps) are discretized to the segment grid by assigning a boundary label to the segment containing each timestamp. The model is trained with binary cross-entropy loss on these labels. This formulation enables direct evaluation under protocol T1 (discrete-time) and straightforward mapping to T2 (continuous-time) by converting predicted segment boundaries back to timestamps.

3.4 Multimodal LLMs

Finally, we evaluate MLLMs to assess whether instruction-following, *general-purpose* models can perform the audio chaptering task. Unlike the previously considered approaches, which focus solely on segmentation, MLLMs can be prompted to jointly perform multiple subtasks end-to-end in a single pass. Our main setting requires joint transcription, segmentation, and chapter title generation. We investigate the following prompting strategies and adaptations.

Zero-Shot. We evaluate the models in a zero-shot setting, prompting them to transcribe alongside chapter boundaries and titles using tags.

Chunking. To mitigate limitations for long audio, we experiment with splitting the audio into 30 second segments, and for each, the model is provided with the audio chunk along with the textual output from previous chunks, and prompted to continue the transcript with chapter annotations.

In-Context Learning (ICL). To further improve task understanding and output formatting, we evaluate in-context learning (ICL; Brown et al. 2020).

Self-Cascaded. To disentangle whether the observed performance of MLLMs arises from their multimodal capabilities or from their underlying language modeling capacity, we evaluate two self-cascaded variants: (1) the model is first prompted to generate a transcript, and in a second step is asked to insert chapters based solely on the transcript; and (2) the model first generates a transcript, and in a second step is provided with both the audio and transcript to insert chapter annotations.

LoRA Training. Finally, we fine-tune the MLLM using LoRA (Hu et al., 2021) to evaluate if lightweight task-specific adaptation improves performance over prompting-based approaches.

3.4.1 Temporal Grounding

In addition to the main setting, we explore two alternative task formulations to investigate whether chapters can be temporally grounded: (1) **Timestamps + Titles**, where the model predicts chapter boundaries and titles directly from audio without generating a transcript. This is a particularly efficient formulation as it avoids generating all transcription tokens while still producing a segmentation; and (2) **Transcription + Timestamps + Titles**, which extends the main setting by additionally requiring timestamps for each chapter boundary, combining transcription, segmentation, temporal grounding, and title generation in a single pass.

4 Experimental Setup

4.1 Datasets

We use YTSeg (Retkowski and Waibel, 2024) as our primary dataset. It contains 19,299 English YouTube videos with their transcripts and chapters. To facilitate more fine-grained analysis of model behavior beyond aggregate metrics, we augment each example with additional annotations. Specifically, every instance is labeled with (i) a *duration regime*, (ii) a *speaker regime*, and (iii) two ASR *transcript variants*. We detail these three annotations in the following and have released them publicly.

Duration Regime. We annotate each recording with a duration regime (Table 2) for two reasons. First, language model performance has been shown to degrade with increasing context length (Liu et al., 2024). Second, duration reflects computational constraints: many systems cannot process long transcripts. This limitation is particularly acute in audio settings, where audio translates to substantially more model tokens than their text counterparts.

Category	Range	Fraction
Short	0–<10 min	38.4%
Medium	10–<30 min	44.2%
Long	30–<60 min	11.1%
Very long	≥60 min	6.3%

Table 2: Duration categories and coverage in the dataset.

Speaker Regime. We add a speaker regime to enable analysis by conversational structure, as multi-speaker audio (e.g., speaker changes and overlap) is generally more challenging for speech processing. Videos are categorized as *Single Speaker*, *Weak Single Speaker*, or *Multi Speaker*, see Table 3 and Appendix H.2 for details on the diarization.

Category	Definition	Fraction
Single Spk.	$N = 1$ **or** $N \geq 2 \wedge p_{\mathrm{d}} \geq 0.95$	74.2%
Weak S. Spk.	$N \geq 2 \wedge 0.8 \leq p_{\mathrm{d}} < 0.95$	13.0%
Multi Spk.	$N \geq 2 \wedge p_{\mathrm{d}} < 0.8$	12.9%

Table 3: Speaker (Spk.) categories based on number of speakers N and dominant speaker proportion p_{d}.

Transcripts. Finally, we transcribe YTSeg, as the dataset provides only reference transcripts (Ref). We use two ASR models with substantially different capacities, Whisper Tiny ($\mathrm{ASR_T}$) and Whisper Large ($\mathrm{ASR_L}$), to study the effect of transcription quality. As expected, $\mathrm{ASR_T}$ yields higher WERs than $\mathrm{ASR_L}$, see Table A17. This impact also extends to sentence segmentation, where $\mathrm{ASR_T}$ produces coarser segmentation with fewer sentences and segments than Ref and $\mathrm{ASR_L}$ (Table A18).

Cross-Domain Generalization. To assess generalization, we additionally evaluate on the AMI corpus (Carletta et al., 2006), an out-of-domain benchmark of multi-party meetings whose segmentation differs substantially from YouTube chaptering.

4.2 Models

Text-Based Models. We evaluate text-based segmentation models that operate on transcripts. As our baseline, we use MiniSeg for which we train variants on different transcripts: reference transcripts (Ref), ASR transcripts from Whisper Tiny ($\mathrm{ASR_T}$) and Whisper Large ($\mathrm{ASR_L}$), and combinations of them. For zero-shot comparison, we evaluate LLaMA 3.1 8B (Grattafiori et al., 2024) with constrained decoding (Retkowski and Waibel, 2025) and WtP[2] (Minixhofer et al., 2023).

[2] 🤗 benjamin/wtp-canine-s-12l

Model	Evaluated on Ref			Evaluated on ASR_T			Evaluated on ASR_L		
	F1 (↑)	**B** (↑)	P_k (↓)	**F1** (↑)	**B** (↑)	P_k (↓)	**F1** (↑)	**B** (↑)	P_k (↓)
Zero-shot (no fine-tuning)									
LLaMA 3.1 8B (Const. Dec.)	25.92	20.69	39.98	24.71	19.91	40.60	26.26	20.78	39.75
$\text{WtP}_{\text{canine-s-12l}}$	28.92	20.52	47.41	28.99	20.58	45.69	28.79	20.39	47.04
Trained on YTSeg (fine-tuned)									
MiniSeg Ref	**39.54**	33.21	30.13	35.87	29.15	32.09	35.58	29.39	32.14
MiniSeg ASR_T	38.40	31.32	31.20	37.30	**30.72**	31.84	36.13	29.54	32.78
MiniSeg ASR_L	37.49	30.74	31.24	35.42	28.78	32.17	35.65	29.52	31.96
MiniSeg Ref+ASR_T	**40.01**	**33.24**	**29.84**	**37.76**	**30.66**	**31.29**	**36.38**	**29.64**	31.97
MiniSeg Ref+ASR_L	39.05	32.18	30.28	35.45	28.54	32.20	35.52	29.06	**31.90**
MiniSeg Ref+$\text{ASR}_\text{T+L}$	**39.54**	**33.01**	**29.96**	**37.38**	30.53	**31.45**	**36.52**	**30.08**	**31.29**

Table 4: Segmentation performance of text-based models in zero-shot settings and after training on the YTSeg dataset. MiniSeg variants are trained on Ref transcripts, ASR_T (Whisper Tiny), ASR_L (Whisper Large), or their combinations, and evaluated on each transcript type.

Text-Based Models with Audio Features. To assess whether acoustic cues improve text-based segmentation, we augment MiniSeg with hand-crafted audio features. We evaluate: (1) audio features only (without text), (2) text with individual feature categories (pauses, speaking rate, pitch, loudness, speakers), and (3) text with all features combined.

Audio-Only Models. We develop AudioSeg, a model that operates directly on frame-level acoustic features without requiring transcripts (Section 3.3). We train and evaluate variants of AudioSeg using different pre-trained audio encoders: HuBERT Base and Large (Hsu et al., 2021), Whisper Large (Radford et al., 2023), AF3-Whisper (Goel et al., 2025), Qwen3-AuT (Xu et al., 2025b), and $\text{PE}_{\text{A-Frame}}$ Base (Vyas et al., 2025).

MLLMs. For MLLMs, we consider Qwen2.5-Omni[3] (Xu et al., 2025a) and Qwen3-Omni[4] (Xu et al., 2025b), using their default inference parameters. Prompts for all variants are provided in Appendix C. For LoRA finetuning, we use Qwen2.5-Omni and the hyperparameters in Table A15.

4.3 Evaluation

We evaluate all models using Protocol **T1 (Discrete-time)** with *6-second chunks*, enabling direct comparison across text-based, audio-only, and multimodal models. This chunk duration corresponds to the average sentence length in YTSeg, providing comparable granularity to text-based segmentation. Text-based models and MLLMs are mapped to the time grid via FA of their (predicted) transcripts to extract sentence start timestamps, while AudioSeg operates directly on 6-second windows. We report three metrics computed on the discretized time grid: F1@6s (↑), B@6s (↑), and P_k@6s (↓).

[3] 🤗 Qwen/Qwen2.5-Omni-7B
[4] 🤗 Qwen/Qwen3-Omni-30B-A3B-Instruct

4.4 Experiments

We design our experiments to address the following research questions:

Q1: *What impact does transcript quality have on text-based segmentation?* We compare segmentation performance using ASR transcripts of varying quality against the reference transcript.

Q2: *Does incorporating audio information improve segmentation?* We explore (i) task-specific models with hand-crafted audio features and learned audio features, and (ii) MLLMs.

Q3: *Beyond segmentation, can MLLMs temporally ground and title chapters?*

Q4: *How do audio characteristics and domain affect model performance?* We analyze performance as a function of the number of speakers and the duration and validate findings on the AMI dataset.

Q5: *How reliable and comparable are different evaluation protocols?* We analyze how transcript choice affects metric comparability and validate time-based evaluation.

5 Results and Analysis

5.1 Q1: What impact does transcript quality have on text-based segmentation?

Table 4 suggests only a weak correspondence between WER and segmentation performance. Zero-shot models (LLaMA 3.1, WtP) that were trained on large corpora demonstrate consistent performance across transcript types, with F1 scores varying by less than 1.5 points. Finetuned MiniSeg models, however, show modest degradation when evaluated on different transcript types than they were trained on. Models trained on reference transcripts lose approximately 3-4 F1 points when evaluated on ASR transcripts. However, joint training on both reference and ASR transcripts substantially improves robustness: MiniSeg Ref+ASR$_\text{T}$ achieves the best F1 score on reference transcripts (40.01) while maintaining strong performance on ASR$_\text{T}$ (37.76) and ASR$_\text{L}$ (36.38). Although Whisper Large achieves lower WER than Whisper Tiny, MiniSeg models trained on ASR$_\text{L}$ perform slightly worse than those on ASR$_\text{T}$ across most conditions. This indicates that ASR quality does not directly translate to segmentation performance, suggesting other factors beyond word-level accuracy are at play.

5.2 Q2: Does incorporating audio information improve segmentation?

Variant (MiniSeg ASR$_\text{T}$)	F1 (↑)	B (↑)	P_k (↓)
Baselines			
Random baseline [F.3]	8.57	7.90	48.43
Audio features only	19.39	14.56	37.85
Text only	37.30	30.72	31.84
Text + single audio feature			
+ Speaking Rate	37.32	30.85	31.75
+ Pitch	36.77	30.35	31.91
+ Loudness	37.82	31.02	31.50
+ Speakers	37.97	31.11	31.48
+ Pauses	**40.17**	**33.59**	**30.25**
Text + all audio features			
Feature Combination	**40.30**	**33.48**	**30.35**

Table 5: Segmentation performance of MiniSeg variants trained on ASR$_\text{T}$ transcripts.

Hand-Crafted Audio Features. Table 5 presents results for MiniSeg trained on ASR$_\text{T}$ transcripts augmented with acoustic features. Audio features alone (F1=19.39) substantially outperform the random baseline (F1=8.57), indicating that acoustic cues carry segmentation-relevant information even without semantics. However, the gap to text-only performance (F1=37.30) confirms that semantic content remains essential. When combined with text, specific feature categories yield notable improvements. Pause duration provides the largest gain (+2.87 F1), while speaker features and loudness offer modest benefits, and pitch and speaking rate show no improvement. Combining all features achieves F1=40.30, indicating that the improvements are primarily driven by pause information rather than contributions from multiple features.

Audio Encoder	F1 (↑)	B (↑)	P_k (↓)
Whisper Large	**45.52**	**36.17**	**28.89**
HuBERT Base	31.07	24.07	34.02
HuBERT Large	35.58	27.95	32.23
AF3-Whisper	**39.02**	**30.75**	**31.23**
Qwen3-AuT	24.66	18.45	35.95
PE$_\text{A-Frame}$ Base	31.36	24.33	33.27

Table 6: Segmentation performance of AudioSeg with different audio encoders.

Model	Strategy[1]	F1 (↑)	B (↑)	P_k (↓)
Qwen2.5-Omni	Default	3.43	2.34	42.67
	ICL	18.86	12.93	40.68
	Chunk	12.14	7.40	53.51
	Self-casc. T	9.49	6.53	42.39
	Self-casc. T+A	17.96	12.52	41.22
	LoRA	24.67	17.44	38.18
Qwen3-Omni	ICL	**41.30**	**35.22**	**33.00**
	Self-casc. T	21.98	18.25	36.90
	Self-casc. T+A	36.14	30.51	**34.66**
Task variant: Segmentation with Timestamps				
Qwen2.5-Omni	ICL -Tr. *(pred)*	0.09	0.15	43.15
	LoRA -Tr. *(pred)*	0.21	0.45	43.40
Qwen3-Omni	ICL -Tr. *(pred)*	12.06	12.17	49.45
	ICL +Tr. *(pred)*	12.52	12.68	46.97
	ICL +Tr. *(FA)*	**43.84**	**37.83**	34.83

[1] T = transcript-only, T+A = transcript+audio.
[2] -Tr. = no transcription, +Tr. = with transcription.
[3] *(pred)*: using model-predicted timestamps
[4] *(FA)*: obtain timestamps via forced alignment

Table 7: Segmentation performance for Qwen Omni models, for videos with duration <30 minutes.

Audio-Only Models. Table 6 demonstrates that learned audio representations can achieve competitive performance without any transcript, though encoder choice critically affects results. Whisper Large achieves the best results (F1=45.52), substantially outperforming all text-based models in Table 4. As an ASR encoder, Whisper implicitly captures linguistic structure alongside acoustic patterns, likely explaining its strong performance, possibly aided by domain match, as it was trained on YouTube data similar to YTSeg. HuBERT Large achieves moderate performance (F1=35.58), while PE$_\text{A-Frame}$ (F1=31.36), trained for sound event detection, indicates that non-speech acoustic cues provide some but insufficient signal. Notably, adapting encoders for MLLM pipelines degrades performance: AF3-Whisper (F1=39.02) underperforms Whisper, and Qwen3-AuT (F1=24.66), trained from scratch for Qwen3-Omni without Whisper's foundation, fares worse still. This likely reflects that such encoders are shaped for consumption by an upstream LLM rather than standalone use with lightweight heads, mirroring similar findings in vision (Zhai et al., 2024; Liu et al., 2026).

MLLMs. Tables A2 and 7 report MLLM performance under various inference strategies. Due to context limitations, these results are restricted to videos <30 minutes. Qwen2.5-Omni exhibits weak instruction following, frequently dropping transcript segments or entering generation loops. Without adaptation (Default), it achieves only F1=3.43, with 62.41% of outputs missing transcripts entirely (Table A2). ICL improves output quality and segmentation (F1=18.86), eliminating missing transcriptions, but the model still produces only 51.61% of the expected transcript length. Qwen3-Omni with ICL demonstrates stronger instruction following (99% of expected length) and achieves F1=41.30. Comparing Self-casc. variants reveals the value of audio: Qwen3-Omni chaptering with both transcript and audio (T+A: F1=36.14) substantially outperforms text-only (T: F1=21.98), demonstrating that audio provides complementary signals. Chunk degrades performance (F1=12.14) due to excessive generation loops, and LoRA of Qwen2.5-Omni improves segmentation (F1=24.67) but introduces severe hallucinations.

Non-Speech Cues. Inspecting the examples where AudioSeg yields the largest gains over text-based models[5], we find a consistent pattern: these videos often contain non-speech audio around chapter transitions, such as music or sound effects. Such cues are only weakly reflected in transcripts, but are directly available to an audio-only model. To confirm this, we filtered out non-speech audio using DeepFilterNet (Schröter et al., 2022), which degraded performance (F1: -2.21, Table A1), validating the importance of these acoustic cues.

5.3 Q3: Beyond segmentation, can MLLMs temporally ground and title chapters?

Temporal Grounding. Table 7 also reports results for two task variants that require the model to predict explicit timestamps. Without transcription (-Tr.), Qwen3-Omni achieves only F1=12.06, and Qwen2.5-Omni fails almost entirely (F1=0.09). Including transcription (+Tr.) barely helps when evaluated on predicted timestamps (F1=12.52), revealing that the bottleneck is temporal grounding rather than segmentation ability. This is confirmed by evaluating the same outputs using FA timestamps instead: F1 jumps to 43.84, surpassing even the main ICL setting (41.30). The predicted timestamps deviate from FA by a median of 11.8 s (mean: 22.0 s), with only 43% falling within 10 s. These results suggest that while MLLMs can identify topical boundaries, they struggle to localize them precisely in time. Generating a transcript remains essential for reliable temporal grounding via FA.

Title Quality. Table A11 reports title quality under two protocols (TM, GC; Appendix B). Both protocols capture complementary aspects. For example, Qwen3-Omni ICL leads on overall quality (GC RL: 26.94) but is comparatively weaker on matched-chapter titles (TM RL: 28.41), whereas Qwen2.5-Omni LoRA substantially lifts the latter (TM RL: 49.99) without a comparable gain in GC.

5.4 Q4: How do audio characteristics and domain affect model performance?

Duration Effects. Figure 2 and Table A3 reveal that AudioSeg clearly dominates for videos <30 minutes, peaking at around 10 minutes. Beyond 20–30 minutes, performance degrades sharply for all models while text-based approaches become competitive again: at ≥60 minutes, MiniSeg + Features (F1=15.17) slightly outperforms AudioSeg (F1=13.28). This convergence at longer durations likely reflects the data distribution: longer videos are underrepresented in training (6.3% of videos are ≥60 min vs. 38.4% for <10 min), and chapter boundaries become substantially sparser (0.170 vs. 1.178 segments/min; see Table A16).

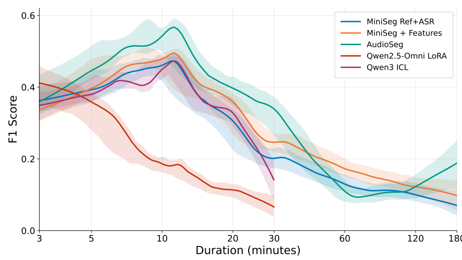

Figure 2: Relation between duration and segmentation performance across models. Smoothed using LOESS.

Speaker Regime Effects. Figure 3 and Table A4 show that multi-speaker content poses challenges for all approaches, with AudioSeg demonstrating the greatest robustness (F1 dropping from 54.62 to 35.64 between single and multi-speaker settings). Notably, while Table 5 shows only modest overall gains from speaker features (+0.67 F1), the breakdown by speaker regime reveals their true value:

[5]E.g., fVFTZu4GaA0, unqhMEgBzck, UBzUJB8cLkI

in multi-speaker videos, MiniSeg + Speakers improves F1 from 26.10 to 29.05 (+2.95), whereas single-speaker content is unaffected.

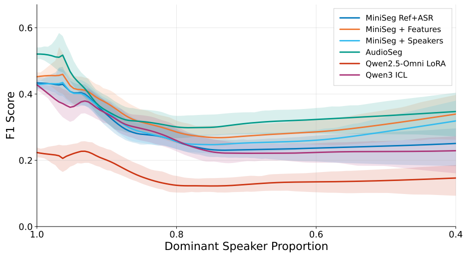

Figure 3: Relation between dominant speaker proportion and segmentation performance across models, for videos <30 minutes. Smoothed using LOESS.

Domain Effects. Tables A5 and A6 show that our main conclusions hold on the AMI dataset: AudioSeg remains the strongest system, and MLLMs do not close the gap. Notably, Qwen3-Omni, which was competitive on YTSeg, performs poorly on AMI, suggesting limited transfer to meeting-style segmentation. Moreover, YTSeg pretraining transfers meaningfully across domains: for both AudioSeg and MiniSeg, the YTSeg→AMI setting consistently outperforms training on AMI alone.

5.5 Q5: How reliable and comparable are different evaluation protocols?

Protocol Comparison. Oracle segmentation (Table A7) quantifies the inherent lossiness of transcript-based protocols: even with perfect boundary selection, they only achieve F1≈73–81%. This discretization loss strongly motivates time-based evaluation protocols that operate directly in continuous time. Protocol comparison (Table A10) reveals that transcript granularity can inflate text-space metrics for identical predictions due to coarser ASR segmentation. Despite this, Protocol T1 maintains ranking consistency with text-space protocols, validating its use for fair, transcript-invariant comparison across modeling paradigms.

Granularity Sensitivity. To assess whether our findings depend on the choice of evaluation granularity, we report T1 scores across multiple chunk sizes (6s, 8s, 10s, 12s) in Table A8. Overall, model rankings are consistent and as expected, F1 generally increases with coarser discretization due to wider tolerance windows. We note that AudioSeg exhibits some non-monotonic variation at 8s and 10s chunks; this reflects a re-discretization artifact rather than a true performance change, as the model predicts on a native 6s resolution, and re-binning into non-aligned chunk sizes can cause boundaries to shift across bin edges or collide. To complement T1, we additionally report T2 collar-based F1 (±3s, ±6s) in Table A9, which operates in continuous time and is therefore free of such discretization artifacts. T2 results show consistent trends and rankings, confirming our conclusions.

6 Related Work

Audio chaptering is still an emerging research area that has predominantly been treated as a text segmentation problem operating over transcripts without considering acoustic features (Ghazimatin et al., 2024; Retkowski and Waibel, 2024). Limited work has explored acoustic features. Earlier work relied on hand-crafted audio features combined with engineered lexical features (Lai et al., 2016; Soares and Barrére, 2018; Lai et al., 2020). More recently, Freisinger et al. (2025) studied hierarchical segmentation of long-form videos but incorporated acoustic information only via pauses. At the same time, recent video chaptering work either relies solely on visual cues or emphasizes combining transcripts with visual information, with little attention to audio (Yang et al., 2023; Ventura et al., 2025).

7 Conclusion

We presented a systematic study of audio chaptering, comparing text-based, audio-only, and multimodal approaches while formalizing evaluation protocols that enable transcript-invariant comparison. Our experiments yield several key findings. First, transcript-free segmentation is not only viable but superior: AudioSeg with Whisper encoders substantially outperforms the best text-based models, demonstrating that learned audio representations capture structural cues beyond what transcripts provide. Second, among acoustic features, pause duration drives nearly all gains when augmenting text-based models, while speaker features provide targeted improvements specifically in multi-speaker content. Third, recent MLLMs such as Qwen3-Omni achieve reasonable ICL performance on shorter audio, though context limitations restrict their applicability. Finally, ASR quality does not directly predict segmentation performance, and joint training on reference and ASR transcripts improves robustness. These findings establish audio chaptering as a task where audio provides complementary or even superior information to text.

8 Limitations

Our experiments primarily rely on a single dataset, YTSeg, though we additionally validate our main findings on the small-scale AMI meeting corpus. While YTSeg contains diverse, real-world audio and video content, this reliance may limit the generalizability of our findings. The dataset is also English-only, which restricts conclusions about multilingual applicability. To the best of our knowledge, however, YTSeg is the only publicly available large-scale dataset for audio chaptering, and no comparable datasets exist in other languages that provide similar scale and annotations for this task. Finally, although it would be interesting to fine-tune strong (and bigger) multimodal foundation models such as Qwen3-Omni for this task, we were unable to do so due to computational constraints. Relatedly, while YTSeg naturally extends into the visual modality, we focus on text and audio only; incorporating vision cues (e.g., scene changes, on-screen text, or slide transitions) may further improve performance and remains an important direction for future work.

9 Potential Risks

We do not foresee significant risks arising from this work. The primary potential issue is incorrect or imprecise audio chaptering, which may lead to user annoyance, reduced usability, or minor inefficiencies when navigating long audio content. In downstream applications, such errors could cause users to miss relevant sections or expend additional effort locating desired information. However, these consequences are limited in scope and reversible, as chaptering outputs can be corrected or refined without permanent impact. We do not anticipate safety-critical or societal harms resulting from the use of the proposed methods.

Acknowledgments

This research is supported by the project "How is AI Changing Science? Research in the Era of Learning Algorithms" (HiAICS), funded by the Volkswagen Foundation, and partially by the European Union's Horizon research and innovation programme under grant agreement No. 101135798, project Meetween (My Personal AI Mediator for Virtual MEETtings BetWEEN People).

References

Doug Beeferman, Adam Berger, and John Lafferty. 1999. Statistical Models for Text Segmentation. *Machine Learning*, 34(1):177–210.

Hervé Bredin. 2017. pyannote.metrics: A toolkit for reproducible evaluation, diagnostic, and error analysis of speaker diarization systems. In *Interspeech 2017*, pages 3587–3591.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Jean Carletta, Simone Ashby, Sebastien Bourban, Mike Flynn, Mael Guillemot, Thomas Hain, Jaroslav Kadlec, Vasilis Karaiskos, Wessel Kraaij, Melissa Kronenthal, Guillaume Lathoud, Mike Lincoln, Agnes Lisowska, Iain McCowan, Wilfried Post, Dennis Reidsma, and Pierre Wellner. 2006. The ami meeting corpus: A pre-announcement. In *Machine Learning for Multimodal Interaction, Second International Workshop*, number 10 in Lecture Notes in Computer Science, pages 28–39, Germany. Springer. 10.1007/11677482_3 ; 2nd International Workshop on Machine Learning for Multimodal Interaction, MLMI 2005 ; Conference date: 11-07-2005 Through 13-07-2005.

Chris Fournier. 2013. Evaluating text segmentation using boundary edit distance. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1702–1712, Sofia, Bulgaria. Association for Computational Linguistics.

Steffen Freisinger, Philipp Seeberger, Thomas Ranzenberger, Tobias Bocklet, and Korbinian Riedhammer. 2025. Towards Multi-Level Transcript Segmentation: LoRA Fine-Tuning for Table-of-Contents Generation. In *Interspeech 2025*, pages 276–280.

Azin Ghazimatin, Ekaterina Garmash, Gustavo Penha, Kristen Sheets, Martin Achenbach, Oguz Semerci, Remi Galvez, Marcus Tannenberg, Sahitya Mantravadi, Divya Narayanan, Ofeliya Kalaydzhyan, Douglas Cole, Ben Carterette, Ann Clifton, Paul N. Bennett, Claudia Hauff, and Mounia Lalmas. 2024. PODTILE: facilitating podcast episode browsing with auto-generated chapters. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management, CIKM 2024, Boise, ID, USA, October 21-25, 2024*, pages 4487–4495. ACM.

Arushi Goel, Sreyan Ghosh, Jaehyeon Kim, Sonal Kumar, Zhifeng Kong, Sang gil Lee, Chao-Han Huck Yang, Ramani Duraiswami, Dinesh Manocha, Rafael Valle, and Bryan Catanzaro. 2025. Audio flamingo

3: Advancing audio intelligence with fully open large audio language models. *Preprint*, arXiv:2507.08128.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Camille Guinaudeau, Guillaume Gravier, and Pascale Sébillot. 2010. Improving asr-based topic segmentation of tv programs with confidence measures and semantic relations. In *Interspeech 2010*, pages 1365–1368.

Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. 2021. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE ACM Trans. Audio Speech Lang. Process.*, 29:3451–3460.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *Preprint*, arXiv:2106.09685.

Nithin Rao Koluguri, Taejin Park, and Boris Ginsburg. 2022. Titanet: Neural model for speaker representation with 1d depth-wise separable convolutions and global context. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2022, Virtual and Singapore, 23-27 May 2022*, pages 8102–8106. IEEE.

Omri Koshorek, Adir Cohen, Noam Mor, Michael Rotman, and Jonathan Berant. 2018. Text Segmentation as a Supervised Learning Task. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 469–473, New Orleans, Louisiana. Association for Computational Linguistics.

Ludwig Kürzinger, Dominik Winkelbauer, Lujun Li, Tobias Watzel, and Gerhard Rigoll. 2020. Ctc-segmentation of large corpora for german end-to-end speech recognition. In *Speech and Computer*, pages 267–278, Cham. Springer International Publishing.

Catherine Lai, Mireia Farrús, and Johanna D. Moore. 2016. Automatic Paragraph Segmentation with Lexical and Prosodic Features. In *Interspeech 2016*, pages 1034–1038. ISSN: 2958-1796.

Catherine Lai, Mireia Farrús, and Johanna D. Moore. 2020. Integrating lexical and prosodic features for automatic paragraph segmentation. *Speech Communication*, 121:44–57.

Xinjian Li. 2023. Low-Resource Speech Recognition for Thousands of Languages.

Chen-Hsuan Liao and Jiun-Yu Wu. 2023. Learning analytics on video-viewing engagement in a flipped statistics course: Relating external video-viewing patterns to internal motivational dynamics and performance. *Comput. Educ.*, 197(C).

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173.

Yikun Liu, Yuan Liu, Shangzhe Di, Haicheng Wang, Zhongyin Zhao, Le Tian, Xiao Zhou, Jie Zhou, Jiangchao Yao, Yanfeng Wang, and Weidi Xie. 2026. Versavit: Enhancing mllm vision backbones via task-guided optimization. *arXiv preprint arXiv:2602.09934*.

Michal Lukasik, Boris Dadachev, Kishore Papineni, and Gonçalo Simões. 2020. Text Segmentation by Cross Segment Attention. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4707–4716, Online. Association for Computational Linguistics.

Annamaria Mesaros, Toni Heittola, and Tuomas Virtanen. 2016. Metrics for polyphonic sound event detection. *Applied Sciences*, 6(6).

Benjamin Minixhofer, Jonas Pfeiffer, and Ivan Vulić. 2023. Where's the point? self-supervised multilingual punctuation-agnostic sentence segmentation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7215–7235, Toronto, Canada. Association for Computational Linguistics.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. 2023. Robust speech recognition via large-scale weak supervision. In *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 28492–28518. PMLR.

Fabian Retkowski and Alexander Waibel. 2024. From text segmentation to smart chaptering: A novel benchmark for structuring video transcriptions. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 406–419, St. Julian's, Malta. Association for Computational Linguistics.

Fabian Retkowski and Alexander Waibel. 2025. Paragraph segmentation revisited: Towards a standard task for structuring speech. *Preprint*, arXiv:2512.24517.

Hendrik Schröter, Alberto N. Escalante-B., Tobias Rosenkranz, and Andreas Maier. 2022. DeepFilterNet: A low complexity speech enhancement framework for full-band audio based on deep filtering.

In *ICASSP 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE.

Eduardo R. Soares and Eduardo Barrére. 2018. Automatic topic segmentation for video lectures using low and high-level audio features. In *Proceedings of the 24th Brazilian Symposium on Multimedia and the Web*, WebMedia '18, page 189–196, New York, NY, USA. Association for Computing Machinery.

Lucas Ventura, Antoine Yang, Cordelia Schmid, and Gül Varol. 2025. Chapter-llama: Efficient chaptering in hour-long videos with llms. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2025, Nashville, TN, USA, June 11-15, 2025*, pages 18947–18958. Computer Vision Foundation / IEEE.

Apoorv Vyas, Heng-Jui Chang, Cheng-Fu Yang, Po-Yao Huang, Luya Gao, Julius Richter, Sanyuan Chen, Matt Le, Piotr Dollár, Christoph Feichtenhofer, Ann Lee, and Wei-Ning Hsu. 2025. Pushing the frontier of audiovisual perception with large-scale multimodal correspondence learning. *Preprint*, arXiv:2512.19687.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Jin Xu, Zhifang Guo, Jinzheng He, Hangrui Hu, Ting He, Shuai Bai, Keqin Chen, Jialin Wang, Yang Fan, Kai Dang, Bin Zhang, Xiong Wang, Yunfei Chu, and Junyang Lin. 2025a. Qwen2.5-omni technical report. *Preprint*, arXiv:2503.20215.

Jin Xu, Zhifang Guo, Hangrui Hu, Yunfei Chu, Xiong Wang, Jinzheng He, Yuxuan Wang, Xian Shi, Ting He, Xinfa Zhu, Yuanjun Lv, Yongqi Wang, Dake Guo, He Wang, Linhan Ma, Pei Zhang, Xinyu Zhang, Hongkun Hao, Zishan Guo, and 19 others. 2025b. Qwen3-omni technical report. *Preprint*, arXiv:2509.17765.

Antoine Yang, Arsha Nagrani, Ivan Laptev, Josef Sivic, and Cordelia Schmid. 2023. Vidchapters-7m: Video chapters at scale. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Ozan Raşit Yürüm, Tuğba Taşkaya-Temizel, and Soner Yıldırım. 2022. The use of video clickstream data to predict university students' test performance: A comprehensive educational data mining approach. *Education and Information Technologies*, 28(5):5209–5240.

Yuexiang Zhai, Shengbang Tong, Xiao Li, Mu Cai, Qing Qu, Yong Jae Lee, and Yi Ma. 2024. Investigating the catastrophic forgetting in multimodal large language model fine-tuning. In *Proceedings of the Conference on Parsimony and Learning*, volume 234 of *Proceedings of Machine Learning Research*, pages 202–227. PMLR.

Ruqing Zhang, Jiafeng Guo, Yixing Fan, Yanyan Lan, and Xueqi Cheng. 2019. Outline generation: Understanding the inherent content structure of documents. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'19, page 745–754, New York, NY, USA. Association for Computing Machinery.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with BERT. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

A Supplementary Results

A.1 Audio Ablation

Audio Encoder	F1 (↑)	B (↑)	P_k (↓)
Whisper Large	45.52	36.17	28.89
+ DeepFilterNet (eval only)	43.31	34.31	29.60
+ DeepFilterNet (trained)	43.98	34.79	29.58

Table A1: Performance of AudioSeg when ablated using DeepFilterNet (Schröter et al., 2022), which filters out music and sound effects while preserving human speech.

A.2 Failure Analysis of MLLMs

Model	Strategy[1]	Failure cases (%)				Transcript (%)	
		No Tags	No Transcr.	Empty Chapter[2]	Gen. Loop[3]	WER	Length
Qwen2.5-Omni	Default	13.12	62.41	11.03	2.17	99.98	8.86
	ICL	15.12	0.00	1.34	2.26	93.98	51.61
	Chunk	18.63	0.08	3.68	80.37	320.57	319.09
	Self-casc. T	8.60	0.25	1.67	1.67	93.02	39.05
	Self-casc. T+A	5.43	0.17	1.42	2.92	93.36	56.26
	LoRA	3.01	0.00	0.25	8.19	1032.60	1077.88
Qwen3-Omni	ICL	0.00	0.00	0.08	1.00	22.32	99.31
	Self-casc. T	0.08	0.00	0.58	3.26	61.92	92.62
	Self-casc. T+A	0.00	0.58	0.00	0.17	31.23	87.04

[1] T = transcript-only, T+A = transcript+audio.
[2] **Empty Chapter**: at least one chapter has no transcript.
[3] **Gen. Loop**: a section title/text appears three times.

Table A2: Failure analysis and transcript-quality metrics for Qwen models under different strategies. We report the rate of four failure modes (missing chapter tags, missing transcription, empty chapters, and generation loops) and transcript quality via WER and relative transcript length. Results are for videos with duration <30 minutes.

A.3 Stratified Results

Duration	Model	F1 (↑)	B (↑)	P_k (↓)
<10 min	MiniSeg Ref+ASR_T	44.06	36.98	30.00
	MiniSeg ASR_T + Features ✨	**44.77**	**38.22**	**29.77**
	AudioSeg (Whisper Large)	**50.01**	**40.72**	**28.08**
	Qwen2.5-Omni LoRA	30.33	24.13	35.47
	Qwen3-Omni ICL	41.80	36.12	35.49
10–<30 min	MiniSeg Ref+ASR_T	40.56	33.17	29.71
	MiniSeg ASR_T + Features ✨	**44.40**	**37.13**	**28.22**
	AudioSeg (Whisper Large)	**51.40**	**41.39**	**26.59**
	Qwen2.5-Omni LoRA	19.52	11.42	40.61
	Qwen3-Omni ICL	40.63	34.41	30.77
30–<60 min	MiniSeg Ref+ASR_T	17.48	11.89	37.04
	MiniSeg ASR_T + Features ✨	**21.89**	**15.90**	**35.93**
	AudioSeg (Whisper Large)	**21.76**	**15.28**	**35.58**
	Qwen3-Omni ICL	12.77	9.83	39.89
≥60 min	MiniSeg Ref+ASR_T	11.76	7.19	39.98
	MiniSeg ASR_T + Features ✨	**15.17**	**9.79**	**38.74**
	AudioSeg (Whisper Large)	**13.28**	**8.66**	**38.10**

Table A3: Segmentation performance across best-performing models by video duration bucket.

Speaker Category	Model	F1 (↑)	B (↑)	P_k (↓)
Single Speaker	MiniSeg Ref+ASR_T	45.40	38.03	28.55
	MiniSeg ASR_T + Speakers	44.57	37.76	29.23
	MiniSeg ASR_T + Features	**47.19**	**40.67**	**27.58**
	AudioSeg (Whisper Large)	**54.62**	**44.99**	**25.48**
	Qwen2.5-Omni LoRA	26.09	18.60	38.12
	Qwen3-Omni ICL	44.40	38.09	31.74
Weak Single Speaker	MiniSeg Ref+ASR_T	34.86	28.37	32.25
	MiniSeg ASR_T + Speakers	35.54	29.80	**31.98**
	MiniSeg ASR_T + Features	**38.80**	**31.55**	**31.50**
	AudioSeg (Whisper Large)	**38.99**	29.40	33.12
	Qwen2.5-Omni LoRA	22.89	15.60	37.72
	Qwen3-Omni ICL	35.55	**30.68**	34.28
Multi Speaker	MiniSeg Ref+ASR_T	26.10	19.77	36.81
	MiniSeg ASR_T + Speakers	29.05	21.16	**35.80**
	MiniSeg ASR_T + Features	**30.89**	**22.08**	36.28
	AudioSeg (Whisper Large)	**35.64**	**26.00**	**33.76**
	Qwen2.5-Omni LoRA	15.34	10.71	39.27
	Qwen3-Omni ICL	24.53	18.70	41.20

Table A4: Segmentation performance by speaker category, for videos with duration <30 minutes.

A.4 Results on AMI Dataset

Model	Pretrain	Val F1 (↑)	Test F1 (↑)
AudioSeg	YTSeg → AMI	**31.19**	**41.91**
	AMI only	**21.12**	**21.72**
MiniSeg	YTSeg → AMI	14.68	18.41
	AMI only	8.42	11.85
Qwen2.5-Omni ICL		9.39	7.95
Qwen3-Omni ICL		4.99	9.16

Table A5: AMI segmentation results for meetings with duration <30 minutes (validation: 9, test: 4).

Model	Pretrain	Val F1 (↑)	Test F1 (↑)
AudioSeg	YTSeg → AMI	**28.19**	**36.41**
	AMI only	**19.85**	**21.96**
MiniSeg	YTSeg → AMI	16.42	19.72
	AMI only	7.16	13.11

Table A6: AMI segmentation results for all meetings (validation: 16, test: 16).

A.5 Evaluation Insights

Transcript	Timestamps	F1 (↑)	B (↑)	P_k (↓)
Native timestamps				
Ref	Closed Captions	**80.93**	**88.16**	**2.35**
ASR_T	Decoder Output	79.78	85.44	3.70
ASR_L	Decoder Output	77.34	83.72	3.25
Forced alignment timestamps				
Ref	CTC Alignment	77.13	83.20	4.05
ASR_T	CTC Alignment	76.95	81.07	5.29
ASR_L	CTC Alignment	73.57	78.58	4.94

Table A7: Oracle segmentation performance ceiling under protocol T1. Oracle places boundaries at the nearest sentence to each true chapter boundary, revealing the inherent discretization loss from projecting continuous-time boundaries onto discrete sentence units.

Duration	Model / System	F1@6s (↑)	F1@8s (↑)	F1@10s (↑)	F1@12s (↑)
<30 min	Qwen2.5-Omni ICL	18.86	20.52	22.08	23.28
	Qwen2.5-Omni LoRA	24.67	27.66	29.63	30.96
	Qwen3-Omni ICL	41.30	44.21	46.33	48.14
	AudioSeg (Whisper Large)	**50.75**	**46.70**	**49.25**	**55.61**
	MiniSeg ASR_T + Features ✨	**44.58**	**47.22**	**49.34**	**51.16**
	MiniSeg Ref+ASR_T	42.22	44.72	46.81	48.40
All	AudioSeg (Whisper Large)	**45.52**	**42.01**	**44.26**	**50.13**
	MiniSeg ASR_T + Features ✨	**40.30**	**42.74**	**44.76**	**46.49**
	MiniSeg Ref+ASR_T	37.76	40.09	41.97	43.42

Table A8: F1 score under the T1 protocol over different time chunk sizes, to assess evaluation granularity sensitivity.

Duration	Model / System	F1@±3s (↑)	F1@±6s (↑)
<30 min	Qwen2.5-Omni ICL	16.22	19.30
	Qwen2.5-Omni LoRA	20.71	24.77
	Qwen3-Omni ICL	41.12	46.65
	AudioSeg (Whisper Large)	**47.87**	**50.90**
	MiniSeg ASR_T + Features ✨	**43.36**	**48.16**
	MiniSeg Ref+ASR_T	41.10	45.26
All	AudioSeg (Whisper Large)	**42.50**	**46.36**
	MiniSeg ASR_T + Features ✨	**38.95**	**43.50**
	MiniSeg Ref+ASR_T	36.42	40.27

Table A9: F1 score under the T2 (continuous time) protocol at ±3s and ±6s tolerances.

Model	Evaluated on Ref		Evaluated on ASR_T			
	R1 (↑)	T1 (↑)	H1 (↑)	H2 (↑)	H3 (↑)	T1 (↑)
MiniSeg Ref	43.37	39.54	39.26	35.99	34.97	35.87
MiniSeg ASR_T	40.34	38.40	42.16	36.42	35.39	37.30
MiniSeg Ref+ASR_T	43.63	40.01	42.35	36.90	35.90	37.76

Table A10: Protocol comparison for MiniSeg variants (F1 score)

Model	Strategy	Temporal Matching BS (↑)	RL (↑)	Match	Global Concat. BS (↑)	RL (↑)
BART†	Fully FT	92.61	48.75	–	91.47	49.95
LLaMA 3.1 8B	Default	87.70	17.38	15.86	84.42	20.77
Qwen2.5-Omni	ICL	**91.82**	**29.02**	9.06	72.59	16.29
	LoRA	**92.95**	**49.99**	12.45	**84.00**	**24.74**
Qwen3-Omni	ICL	89.89	28.41	28.45	**86.76**	**26.94**

† From Retkowski and Waibel (2024); uses oracle segmentation; trained and evaluated on reference transcripts.

Table A11: Title quality under the Temporal Matching (TM) and Global Concatenation (GC) protocols, for videos with duration <30 minutes. BS = BERTScore F1, RL = ROUGE-L F1, Match = % of chapters temporally matched.

B Title Quality

B.1 Title Evaluation Protocols

Most work on text segmentation evaluates models solely via boundary quality, even in datasets with section titles (Koshorek et al., 2018; Lukasik et al., 2020). When considered, this is typically done in cascaded pipelines where titles are generated from reference segments, enabling separate evaluation of segmentation and title quality (Retkowski and Waibel, 2024). For end-to-end models that jointly predict boundaries and titles, comparison becomes non-trivial because reference titles are tied to the reference segmentation while predicted titles follow the predicted one. Prior work evaluates titles only when the predicted structure matches the reference (Zhang et al., 2019). In our setting, boundaries lie in continuous time, so exact-match (EM) is not expected even for qualitatively correct predictions. We therefore propose two protocols that specify how titles are compared. Each protocol induces a set of text pairs to which a text similarity metric $m(\cdot,\cdot)$, such as ROUGE (Lin, 2004) or BERTScore (Zhang et al., 2020), can be applied.

B.1.1 Temporally Matched (TM)

TM is the temporal analogue of EM. Let (t_s, t_e, ℓ) be a reference chapter and $(\hat{t}_s, \hat{t}_e, \hat{\ell})$ the prediction, where (t_s, t_e) is the time interval and ℓ the title. A predicted chapter is matched to a reference chapter if boundaries agree within δ seconds:

$$\left|\hat{t}_s - t_s\right| \leq \delta \;\; \wedge \;\; \left|\hat{t}_e - t_e\right| \leq \delta. \qquad (1)$$

We compute $m(\hat{\ell}, \ell)$ for each matched pair and average across matched chapters; unmatched chapters are excluded. TM therefore measures title quality on approximately correct temporal structure.

B.1.2 Global Concatenation (GC)

GC is structure-independent and evaluates titles at the document level. Let $C(\ell_{1:n})$ denote chronological concatenation with newline separators:

$$C(\ell_{1:n}) \;=\; \ell_1 \,\|\, \texttt{\textbackslash n} \,\|\, \cdots \,\|\, \texttt{\textbackslash n} \,\|\, \ell_n. \qquad (2)$$

We then compute $m\big(C(\ell^{p}_{1:L}),\, C(\ell^{g}_{1:K})\big)$. GC is well-defined for $L \neq K$ and captures the collective quality of predicted titles.

C Prompts

LLaMA 3.1 8B System Prompt (Chaptering)

```
You are an AI assistant that helps users organize text into high-level chapters with descriptive titles.
```

LLaMA 3.1 8B User Prompt (Chaptering)

```
You are tasked with organizing a given text into high-level chapters or sections, each with a descriptive title. The text will be
↪ provided to you, and your job is to break it up into a small number of coherent chapters (typically 9 ± 5 chapters per document).
↪ Here's the text you'll be working with:

{input}

Your task is to insert chapter breaks with titles into this text. Each chapter should be introduced with a Markdown-style heading in the
↪ format: \n\n# Chapter Title\n\n

A chapter is a high-level thematic or functional segment of text that encompasses multiple related ideas or a major narrative/conceptual
↪ arc. To identify where to insert chapter breaks and what titles to use, consider the following guidelines:

1. Look for major shifts in topic, theme, or narrative direction
2. Identify distinct phases or stages in the content (e.g., introduction, conclusion)
3. Recognize major transitions between different conceptual areas
4. Consider natural breakpoints that would help a reader understand the overall structure
5. Create descriptive titles that capture the essence of each chapter's content
6. Aim for 9 ± 5 chapters total - fewer, more substantial divisions rather than many small ones

Chapter titles should be:
- Concise but descriptive (2-6 words typically)
- Reflective of the main theme or purpose of that section
- Formatted as proper titles (capitalize important words)

Examples of good chapter titles:
- "Introduction"
- "Historical Background"
- "Main Arguments"
- "Case Study Analysis"
- "Implications"
- "Conclusion"

Please provide your final output with the inserted chapter breaks and titles. Ensure that you maintain the original text exactly as it
↪ was given, only adding the chapter headers where appropriate.
```

Figure A1: LLaMA 3.1 8B system and user prompts for transcript chaptering.

Qwen System Prompt (Chaptering)

```
You are Qwen, a virtual human developed by the Qwen Team, Alibaba Group, capable of perceiving auditory and visual inputs, as well as
↪ generating text and speech. Only return the answer requested. Do not include any explanation or introductions.
```

Qwen System Prompt (Transcription)

```
You are Qwen, a virtual human developed by the Qwen Team, Alibaba Group, capable of perceiving auditory and visual inputs, as well as
↪ generating text and speech. Only return the answer requested. Do not include any explanation or introductions. You are a speech
↪ recognition model.
```

Qwen User Prompt (Transcription)

```
Transcribe the English audio into text.
```

Figure A2: Qwen system prompts and transcription prompt.

Qwen User Audio Chaptering Prompt

```
You are tasked with organizing a given speech recording into
↪ high-level chapters or sections, each with a descriptive
↪ title. Your job is to transcribe the speech and break it up
↪ into a small number of coherent chapters (typically 9 ± 5
↪ chapters per document).

Each chapter should be introduced in the transcript, but the
↪ chapter title must be wrapped in markers for extraction, in
↪ the format:

[CSTART] Chapter Title [CEND]

A chapter is a high-level thematic or functional segment of text
↪ that encompasses multiple related ideas or a major
↪ narrative/conceptual arc. To identify where to insert chapter
↪ breaks and what titles to use, consider the following
↪ guidelines:

1. Look for major shifts in topic, theme, or narrative direction
2. Identify distinct phases or stages in the content (e.g.,
↪ introduction, conclusion)
3. Recognize major transitions between different conceptual areas
4. Consider natural breakpoints that would help a reader
↪ understand the overall structure
5. Create descriptive titles that capture the essence of each
↪ chapter's content
6. Aim for 9 ± 5 chapters total - fewer, more substantial
↪ divisions rather than many small ones

Chapter titles should be:
- Concise but descriptive (1-6 words typically)
- Reflective of the main theme or purpose
- Formatted as proper titles (capitalize important words)

Please provide the final transcript with the inserted chapter
↪ breaks and titles using the [CSTART] … [CEND] format for all
↪ chapter headings.
```

...with In-context Example

```
You are tasked with organizing a given speech recording into
↪ high-level chapters or sections, each with a descriptive
↪ title. Your job is to transcribe the speech and break it up
↪ into a small number of coherent chapters (typically 9 ± 5
↪ chapters per document).

Each chapter should be introduced in the transcript, but the
↪ chapter title must be wrapped in markers for extraction, in
↪ the format:

[CSTART] Chapter Title [CEND]

A chapter is a high-level thematic or functional segment of text
↪ that encompasses multiple related ideas or a major
↪ narrative/conceptual arc. To identify where to insert chapter
↪ breaks and what titles to use, consider the following
↪ guidelines:

1. Look for major shifts in topic, theme, or narrative direction
2. Identify distinct phases or stages in the content (e.g.,
↪ introduction, conclusion)
3. Recognize major transitions between different conceptual areas
4. Consider natural breakpoints that would help a reader
↪ understand the overall structure
5. Create descriptive titles that capture the essence of each
↪ chapter's content
6. Aim for 9 ± 5 chapters total - fewer, more substantial
↪ divisions rather than many small ones

Chapter titles should be:
- Concise but descriptive (1-6 words typically)
- Reflective of the main theme or purpose
- Formatted as proper titles (capitalize important words)

Below is an example of how the beginning of a transcript should
↪ look with chapter titles inserted using this format:
[CSTART] Introduction [CEND] Welcome everyone, and thank you for
↪ joining us today. In this session, we will explore the topic
↪ of automatic audio chaptering. [CSTART] Early Background
↪ [CEND] Before we dive into the main topic, it's important to
↪ understand how the project began...

Please provide the final transcript with the inserted chapter
↪ breaks and titles using the [CSTART] … [CEND] format for all
↪ chapter headings.
```

Figure A3: Qwen user audio chaptering prompts.

Qwen User Audio Chaptering Prompt with Chunked Audio Input

```
You are tasked with transcribing an audio chunk and appending it to a given transcript that serves as context. Moreover, the transcribed
↪ speech should be broken up into a small number of coherent chapters (typically 9 ± 5 chapters per document).

Each chapter should be introduced in the transcript, but the chapter title must be wrapped in markers for extraction, in the format:

[CSTART] Chapter Title [CEND]

A chapter is a high-level thematic or functional segment of text that encompasses multiple related ideas or a major narrative/conceptual
↪ arc. To identify where to insert chapter breaks and what titles to use, consider the following guidelines:

1. Look for major shifts in topic, theme, or narrative direction
2. Identify distinct phases or stages in the content (e.g., introduction, conclusion)
3. Recognize major transitions between different conceptual areas
4. Consider natural breakpoints that would help a reader understand the overall structure
5. Create descriptive titles that capture the essence of each chapter's content
6. Aim for 9 ± 5 chapters total - fewer, more substantial divisions rather than many small ones

Chapter titles should be:
- Concise but descriptive (1-6 words typically)
- Reflective of the main theme or purpose
- Formatted as proper titles (capitalize important words)

Below is an example of how the beginning of a transcript should look with chapter titles inserted using this format:
=== TRANSCRIPT SO FAR (DO NOT REPEAT) ===
[CSTART] Introduction [CEND] Welcome everyone, and thank you for joining us today. This is a very cool educational video and we are
↪ delighted that you are watching it.
=== END OF TRANSCRIPT ===
Continue the transcript using ONLY the new audio chunk.\nInsert chapter titles using [CSTART] ... [CEND] as needed.\nNever repeat or
↪ restate text from the TRANSCRIPT SO FAR.
Output:
In this session, we will explore the topic of automatic audio chaptering. [CSTART] Early Background [CEND] Before we dive into the main
↪ topic, it's important to understand how the project began...

You will receive one audio chunk at the time. Continue the transcript using ONLY the new audio chunk.
=== TRANSCRIPT SO FAR (DO NOT REPEAT) ===
<transcript of previous chunk>
=== END OF TRANSCRIPT ===
Continue the transcript using ONLY the new audio chunk.
Insert chapter titles using [CSTART] ... [CEND] as needed.
Never repeat or restate text from the TRANSCRIPT SO FAR.
```

Figure A4: Qwen user audio chaptering prompt with chunked input audio (30 s chunks).

Qwen User Audio Chaptering Prompt (Transcript)

```
You are tasked with organizing a given transcript into high-level
↪ chapters or sections, each with a descriptive title. Your job
↪ is to use the transcription and insert a small number of
↪ coherent chapters (typically 9 ± 5 chapters per document).

Each chapter should be introduced in the transcript, but the
↪ chapter title must be wrapped in markers for extraction, in
↪ the format:

[CSTART] Chapter Title [CEND]

A chapter is a high-level thematic or functional segment of text
↪ that encompasses multiple related ideas or a major
↪ narrative/conceptual arc. To identify where to insert chapter
↪ breaks and what titles to use, consider the following
↪ guidelines:

1. Look for major shifts in topic, theme, or narrative direction
2. Identify distinct phases or stages in the content (e.g.,
↪ introduction, conclusion)
3. Recognize major transitions between different conceptual areas
4. Consider natural breakpoints that would help a reader
↪ understand the overall structure
5. Create descriptive titles that capture the essence of each
↪ chapter's content
6. Aim for 9 ± 5 chapters total - fewer, more substantial
↪ divisions rather than many small ones

Chapter titles should be:
- Concise but descriptive (1-6 words typically)
- Reflective of the main theme or purpose
- Formatted as proper titles (capitalize important words)

Below is an example of how the beginning of a transcript should
↪ look with chapter titles inserted using this format:
[CSTART] Introduction [CEND] Welcome everyone, and thank you for
↪ joining us today. In this session, we will explore the topic
↪ of automatic audio chaptering. [CSTART] Early Background
↪ [CEND] Before we dive into the main topic, it's important to
↪ understand how the project began...

Please provide the final transcript with the inserted chapter
↪ breaks and titles using the [CSTART] … [CEND] format for all
↪ chapter headings. Make sure you DO NOT change the transcript
↪ itself, only insert chapters.
```

Qwen User Audio Chaptering Prompt (Transcript + Audio)

```
You are tasked with organizing a given speech recording and
↪ corresponding transcript into high-level chapters or sections,
↪ each with a descriptive title. Your job is to use the
↪ transcription and recording and insert a small number of
↪ coherent chapters (typically 9 ± 5 chapters per document).
If the transcript is incorrect at some parts, you can correct it
↪ while inserting the chapters.

Each chapter should be introduced in the transcript, but the
↪ chapter title must be wrapped in markers for extraction, in
↪ the format:

[CSTART] Chapter Title [CEND]

A chapter is a high-level thematic or functional segment of text
↪ that encompasses multiple related ideas or a major
↪ narrative/conceptual arc. To identify where to insert chapter
↪ breaks and what titles to use, consider the following
↪ guidelines:

1. Look for major shifts in topic, theme, or narrative direction
2. Identify distinct phases or stages in the content (e.g.,
↪ introduction, conclusion)
3. Recognize major transitions between different conceptual areas
4. Consider natural breakpoints that would help a reader
↪ understand the overall structure
5. Create descriptive titles that capture the essence of each
↪ chapter's content
6. Aim for 9 ± 5 chapters total - fewer, more substantial
↪ divisions rather than many small ones

Chapter titles should be:
- Concise but descriptive (1-6 words typically)
- Reflective of the main theme or purpose
- Formatted as proper titles (capitalize important words)

Below is an example of how the beginning of a transcript should
↪ look with chapter titles inserted using this format:
[CSTART] Introduction [CEND] Welcome everyone, and thank you for
↪ joining us today. In this session, we will explore the topic
↪ of automatic audio chaptering. [CSTART] Early Background
↪ [CEND] Before we dive into the main topic, it's important to
↪ understand how the project began...

Please provide the final transcript with the inserted chapter
↪ breaks and titles using the [CSTART] … [CEND] format for all
↪ chapter headings.
```

Figure A5: Qwen user audio chaptering prompts using transcript-only vs. transcript+audio input.

D Hyperparameters

Hyperparameter	Value
Architecture	
Sentence Encoder	🤗 `all-MiniLM-L12-v2`
Document Encoder	RoFormer
└ Attention Heads	8
└ Layers	12
└ Embedding Dim	384
Training	
Loss Function	Weighted BCE
Cross-Entropy Weights	$[1, 2]$
Learning Rate	2.5×10^{-5}
Batch Size	115,000 Tokens
Epochs	15
LR Schedule	Cosine
Optimizer	AdamW
Dropout Rate	0.1
Gradient Sampling Rate	0.5

Table A13: Hyperparameters for the architecture and training of MiniSeg

Hyperparameter	Value
LoRA Configuration	
Rank	16
Target Modules	All
Training	
Cutoff Length	32,768
Batch Size (per device)	1
Gradient Acc. Steps	4
Learning Rate	1.0×10^{-5}
Epochs	3
LR Schedule	Cosine
Warmup Ratio	0.1
Precision	FP16
Early Stopping	
Evaluation Strategy	Steps
Patience	100 Steps
Metric	Loss

Table A15: Hyperparameters for LoRA (Hu et al., 2021) finetuning of Qwen 2.5-Omni. We train it for 30 hours on three NVIDIA RTX PRO 6000 Blackwell.

Hyperparameter	Value
Architecture	
Audio Encoder	🤗 `whisper-large-v3`
└ Chunk Size	30 s
└ Batch Chunks	48
Local Seg. Transformer	Transformer
└ Attention Heads	4
└ Layers	3
└ Embedding Dim	384
Document Encoder	RoFormer
└ Attention Heads	8
└ Layers	12
└ Embedding Dim	384
Training	
Loss Function	Weighted BCE
Cross-Entropy Weights	$[1, 2]$
Learning Rate	2.5×10^{-5}
Batch Size	5 h
Chunk Size	6 s
Epochs	5
LR Schedule	Cosine
Optimizer	AdamW
Dropout Rate	0.1
Gradient Sampling Rate	0.5

Table A14: Hyperparameters for the architecture and training of AudioSeg

E Forced Alignment

Forced alignment plays a central role in our pipeline, particularly for sentence-level feature extraction. Computing features at the sentence level requires precise start and end timestamps for each sentence. To this end, we employ CTC-based forced alignment using ALQAlign[6] (Kürzinger et al., 2020; Li, 2023). Accurate temporal alignment is further essential for our evaluation protocol H3, which assesses time-based overlap between sentences.

F Evaluation

F.1 Segmentation Evaluation

We employ the `segeval`[7] library (Fournier, 2013) to compute segmentation evaluation metrics, including P_k and Boundary Similarity. Although our primary evaluation protocol (T1) operates on time-based chunks rather than individual sentences, these chunks are selected to match the average sentence length. Consequently, we retain the default parameter configurations for both metrics.

F.2 Alignment Procedures for H2 and H3

This appendix details the projection mechanisms used in protocols H2 and H3 to map ASR-based boundary predictions onto the reference transcript $S_{\text{ref}} = (s_1, \ldots, s_N)$.

F.2.1 H2: Word-Level Alignment

Protocol H2 derives a monotonic mapping from ASR sentences $S_{\text{asr}} = (u_1, \ldots, u_M)$ to reference sentences via token-level alignment. For each ASR sentence u_j, we identify the reference sentence s_i that contains the majority of its aligned tokens.

A predicted boundary between adjacent ASR sentences u_j and u_{j+1} is projected as follows. If u_j maps to s_i and u_{j+1} maps to $s_{i'}$ with $i' > i$, the boundary is placed at index i in S_{ref}. If both u_j and u_{j+1} map to the same reference sentence s_i, a length-based heuristic assigns the boundary to the nearest sentence edge (i.e., $i-1$ or i, depending on the relative position of the mapped tokens within s_i). This procedure yields a predicted boundary sequence $\hat{\mathbf{y}}$ over S_{ref}, which is then compared to the projected gold boundaries using standard segmentation metrics.

F.2.2 H3: Time-Based Alignment

Protocol H3 uses temporal overlap rather than lexical alignment. Given timestamps for both S_{asr} and S_{ref} (obtained via ASR decoder output or forced alignment), each ASR sentence u_j is mapped to the reference sentence s_i with which it shares the greatest temporal overlap. In case of zero overlap, u_j is assigned to the reference sentence whose midpoint is nearest to its own. The boundary projection logic then follows H2: boundaries between ASR sentences are transferred to S_{ref} based on the established index mapping.

F.3 Random Chaptering Baseline

This baseline generates random segmentation predictions while preserving the total number of segment boundaries from the reference. For each document, it counts how many boundaries (1s) appear in the reference sequence, then randomly distributes exactly that many boundaries across all possible sentence positions.

G Feature Extraction Details

This appendix provides a description of the extraction methodology for the hand-crafted audio and speaker features listed in Table 1.

G.1 Features

Pauses. The `pause_duration` between consecutive sentences was computed as the time gap between the end timestamp of sentence $i-1$ and the start timestamp of sentence i. For the first sentence in each video ($i = 0$), the pause duration was defined as 0. Negative gaps were clipped to 0 to represent zero pause duration.

Speaking Rate. Words per minute (WPM) was calculated for each sentence using whitespace tokenization (`wpm`) by dividing the number of tokens by the sentence duration. To normalize for variability in speaking rates between videos, we additionally computed video-adapted z-scores (`z_wpm`).

Pitch. Fundamental frequency (F_0) features were extracted using the YIN algorithm as implemented in `torchaudio`[8] with a 10 ms frame step, 50–1100 Hz frequency range, and 30-frame median smoothing window. For each sentence with at least 3 voiced frames, we computed absolute features (`mean_f0`, `std_f0`, `min_f0`, `max_f0`, `range_f0`, `voicing_ratio`) and a pitch slope (`slope_f0`)

[6] https://github.com/xinjli/alqalign
[7] https://segeval.readthedocs.io/

[8] torchaudio.functional.detect_pitch_frequency

Duration Category	Videos %	Seg. Duration	Seg. Count	Segs/Min	Speaker Count	Dom. Spk. Prop.
0–<10 min	38.4%	56.0s (0.9m)	7.1	1.178	1.32	0.958
10–<30 min	44.2%	94.6s (1.6m)	9.7	0.681	1.46	0.933
30–<60 min	11.1%	282.1s (4.7m)	9.4	0.221	1.64	0.883
≥60 min	6.3%	360.4s (6.0m)	16.0	0.170	1.56	0.873

Table A16: Video duration categories with segment and speaker statistics.

via linear regression over time. To account for video-specific pitch ranges, we computed a video-level baseline using the median and standard deviation of F_0 across all voiced frames within sentence boundaries, then derived normalized features (`zF0_mean`, `zF0_slope`). Sentences with fewer than 3 voiced frames were assigned zero values.

Loudness. Loudness features were computed using the ITU-R BS.1770-4 standard as implemented in `torchaudio`[9], which measures integrated loudness in LKFS. For each sentence, we extracted the corresponding audio segment and computed its loudness (`lkfs`). Sentences shorter than 0.4 s were zero-padded before computation. To normalize for video-specific recording levels, we computed video-adapted z-scores (`z_loudness`).

Speaker Features. Speaker diarization was performed using a TitaNet-based pipeline (Koluguri et al., 2022), which produces timestamped speaker segments. To assign a single speaker label to each sentence, we computed the temporal overlap between the sentence interval and all diarization segments. The speaker with the maximum total overlap duration was assigned to the sentence. In cases where a sentence was fully contained within a single diarization segment, that speaker was assigned directly. For sentences with no overlapping segments, we used the previous sentence's speaker label as a fallback (or the nearest segment by temporal distance for the first sentence). From these per-sentence speaker assignments, we derived seven features: binary flags for speaker continuation (`same_as_prev`) and speaker changes (`speaker_change`). The `turn_id` feature was computed as the cumulative count of speaker changes. Within-turn features included the 0-indexed position within the current turn (`pos_in_turn`), the total number of sentences in the current turn (`turn_len`), and the distance in sentences since the last occurrence of the same speaker (`dist_prev_same`). Finally, `num_speakers_so_far` tracked the cumulative count of unique speakers encountered from the beginning of the video.

[9]`torchaudio.functional.loudness`

G.2 Feature Normalization

We apply global normalization to all features using z-scores computed on the training set:

$$x_i' = \frac{x_i - \mu_{\text{train}}}{\sigma_{\text{train}}} \quad (3)$$

where μ_{train} and σ_{train} are computed from the training partition and applied to all splits. Features already z-scored within documents (`z_wpm`, `zF0_*`) are not re-normalized.

H Data Annotations

H.1 Duration Categories

Table A16 provides per-category statistics on segment structure and speaker composition. Beyond the training set imbalance and boundary sparsity discussed in Section 5.4, longer videos also show more complex speaker structure: the average number of speakers rises from 1.32 to 1.56 and the dominant speaker proportion decreases from 0.958 to 0.873, indicating a shift toward multi-speaker content such as podcasts and interviews.

H.2 Speaker Diarization

We obtain speaker segments using a TitaNet-based diarization pipeline (Koluguri et al., 2022), then define N as the number of speakers with $\geq$10 seconds of total speech and p_{d} as the dominant speaker's share of total speech time. Based on (N, p_{d}), we are able to categorize the videos into single and multi-speaker settings.

H.3 Data Statistics for Transcripts

Model	Train	Val	Test
Whisper Tiny	20.17%	20.28%	19.88%
Whisper Large	12.81%	13.17%	12.44%

Table A17: WER of Whisper ASR transcripts on YTSeg

	# Segments	# Sentences
ASR (Whisper Tiny)	8.95 ± 6.45	182.79 ± 220.62
ASR (Whisper Large)	9.13 ± 6.73	200.48 ± 282.06
Reference	9.18 ± 6.75	200.23 ± 287.67

Table A18: Data statistics on the YTSeg test dataset when applying sentence segmentation on the ASR transcripts versus the reference transcripts.