PIANOBENCH: EVALUATING MUSICAL PERCEPTION AND REASONING IN LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Multimodal language models continue to improve on broad music understanding benchmarks, but which musical features can they perceive and reason about? We investigate this question by introducing *PianoBench*, a benchmark that tasks language models with reasoning about musical performances across congruent acoustic and symbolic input representations. PianoBench is derived from our new multimodal, training-scale dataset comprising more than 20,000 hours of interleaved transcripts of music and spoken musical commentary from instructional piano videos. On PianoBench, none of the open-weight audio-language models we evaluate significantly outperform a no-audio control, while proprietary frontier models show mixed gains from audio. In contrast, frontier models using chain of thought achieve up to 93.2% accuracy when provided transcripts of music as text, exposing a gap between *reasoning* about musical events in symbolic form and *perceiving* them from audio. To investigate this further, we mid-train `Qwen3.5-9B-Base` on our interleaved dataset to improve use of music transcripts without chain of thought, raising accuracy on PianoBench from 42.8% to 56.0% after instruction tuning. These experiments suggest that instructional videos may represent an overlooked source of multimodal supervision for music. Supplement: https://anon834910.github.io/anon-dataset-demo/.

1 INTRODUCTION

Multimodal language models are increasingly capable of processing inputs beyond text, including images, speech, and general audio (Alayrac et al., 2022; Kong et al., 2024; Xu et al., 2025; Gemma Team et al., 2026). However, it remains unclear which aspects of music these systems actually understand. Open-weight models such as the Audio Flamingo series (Kong et al., 2024; Ghosh et al., 2025b; 2026) and Music Flamingo (Ghosh et al., 2025a) perform strongly on broad music understanding benchmarks (Sakshi et al., 2024; Kumar et al., 2025; Weck et al., 2024), but much of the supervision behind both training and evaluation originates from track-level annotations (e.g., captions, tags). In this work, we examine whether models can understand and articulate more nuanced, localized aspects of music such as rhythmic motifs and chord progressions. This understanding likely requires supervision to relate localized musical patterns with verbal descriptions and analysis. However it is not obvious that this supervision arises naturally from standard data recipes.

One underexplored setting in which language and music naturally coexist is musical instruction. In piano tutorial videos, a teacher introduces an idea, demonstrates it at the keyboard, and comments on what was just played, so that explanations are temporally adjacent to the musical passages they describe. We use an LLM-guided crawl to collect instructional piano videos at scale, then apply source separation (Défossez, 2022; Solovyev et al., 2024) together with speech and piano transcription (Sekoyan et al., 2025; Bradshaw & Colton, 2025; Benetos et al., 2018) to convert audio into timestamped commentary text and transcribed music (Figure 1). Our resulting dataset, **PianoTalk**, comprises 138k videos and more than 20,000 hours of interleaved multilingual commentary and music transcripts, in a text format suitable for standard self-supervised LM training.

From the test split of PianoTalk, we construct **PianoBench**, an evaluation benchmark for nuanced music understanding consisting of 250 human-verified multiple choice questions (MCQs). The MCQs in PianoBench reference the musical content in short piano passages from 162 videos and range in topic from note and chord details to passage-level ideas such as accompaniment and style.

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.

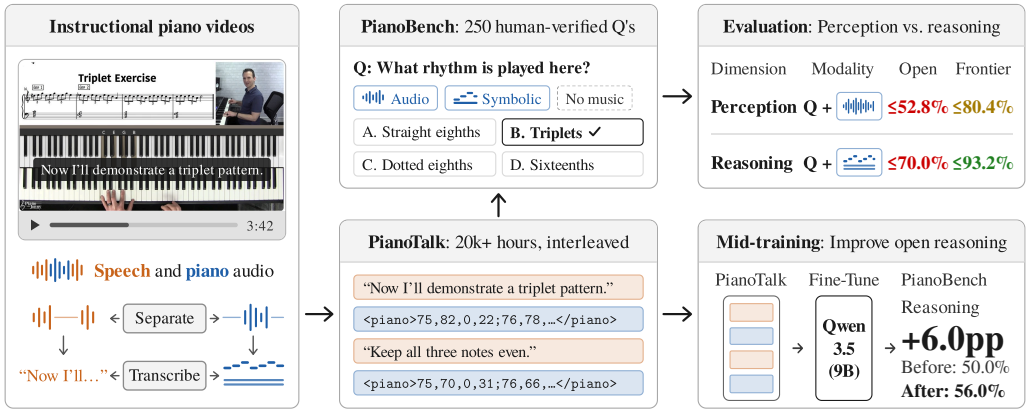


Figure 1: We release PianoBench, a benchmark for evaluating music perception and reasoning capabilities in LLMs. PianoBench is derived from PianoTalk, a training-scale (20k hours) dataset of interleaved commentary and music transcripts, which is itself derived from instructional piano videos. PianoBench exposes two gaps in music understanding: (1) frontier LLMs struggle with perception, performing worse when prompted with audio vs. symbolic transcripts, and (2) open LLMs struggle with reasoning even when given transcripts. We show that mid-training on PianoTalk can shrink the reasoning gap in open models, improving accuracy of `Qwen3.5-9B` by 6.0pp.

Because every passage has both a recording and a transcript, we can represent the same musical content as either piano audio, or as a text-based encoding of the symbolic transcript. The audio condition probes *perception*, i.e., whether a model can recover musical events from sound and use them to answer. The text-based transcript format examines *reasoning*, i.e., whether a model can answer when those events are given explicitly. Following Zang et al. (2025), a no-music control measures the extent to which answers can be gamed from clues in questions alone.

Evaluating open-weight and proprietary audio-language models and text-only LMs on PianoBench reveals two gaps in multimodal music understanding capabilities. The first is a *perception gap*: open and frontier models alike understand music far better when it is represented in text. The best text-based accuracy is 93.2%, compared with 80.4% for the best audio result. All frontier audio LMs we evaluate score 8.8–31.2 percentage points (pp) higher with text as input rather than with the audio. The second is a *reasoning gap*: when note events are explicit, some text-only models answer more questions correctly when using chain of thought (COT) to bring together musical concepts.

To test whether instructional videos can help close the reasoning gap in open models, we mid-train (i.e., continue pretraining) `Qwen3.5-9B-Base` on PianoTalk. The text-only transcripts in PianoTalk can be used for training a standard LM without architectural changes. As a control, we train on a version of PianoTalk separated into non-interleaved documents containing either commentary or music, but not both. We follow both runs with general-purpose instruction tuning that contains no music data. The interleaved model reaches 56.0% on PianoBench, surpassing the official instruction-tuned checkpoint (50.0%) without COT. In contrast, the separated model reaches only 42.4%. It follows that preserving the temporal alignment between commentary and music is useful, suggesting that instructional videos are a promising source of self-supervised multimodal training data for music.

In summary, our contributions are as follows:

1. **Multimodal datasets from musical instruction.** We introduce PianoTalk, a training-scale dataset of interleaved commentary and piano transcripts derived from instructional videos, and PianoBench, a human-verified benchmark that presents musical passages as either audio or transcripts. We will release both upon publication and open source the data pipeline.
2. **Exposing gaps in music understanding.** Through evaluations on PianoBench, we identify a perception gap in frontier models, which reason far better over transcripts than audio, and a reasoning gap in open models, which lag behind frontier models.
3. **Learning from interleaved instruction.** We show that mid-training on PianoTalk in an interleaved format improves an open model's zero-shot use of note transcripts without COT.

2 Related Work

Our work most closely relates to families of work around evaluating the musical understanding capabilities of LMs, as well as work on documented data recipes for training these capabilities.

2.1 Benchmarks for Music Understanding

As music understanding capabilities in language models have progressed, there has been an increase in benchmarks with questions evaluating music knowledge, perception, and reasoning abilities. These typically take the form of multiple-choice questions (MCQs), conditioned on audio or symbolic musical material such as MIDI and scores.

Among commonly used audio understanding benchmarks, MMAU (Sakshi et al., 2024), MMAU-Pro (Kumar et al., 2025), and MMAR (Ma et al., 2025) all include music questions. MuChoMusic (Weck et al., 2024) is a music-only MCQ audio benchmark whose questions were automatically generated from human-written captions, but verified by humans. RUListening (Zang et al., 2025) identified question and answer-option priors that models could exploit in MuChoMusic and proposed text-only distractor scoring, which inspires our approach to PianoBench. HumMusQA (Weck et al., 2026) is a similar music-only benchmark that instead has experts generate questions and evaluate distractors. At a more granular level, MUSE (Carone et al., 2025) tests controlled music perception, MusicTheoryBench (Yuan et al., 2024) uses ABC notation, and MusICA-MetaBench (Sourada et al., 2026) compares performance across audio, ABC notation, and scores. PianoBench differs by drawing questions from commentary in musical instruction videos, spanning note and chord details through passage-level ideas such as style. Matched audio and symbolic inputs let us compare answers based on sound with answers based on explicit note events.

2.2 Training Data and Methods for Music Understanding

The Audio Flamingo (AF) models (1, 2, and 3) (Kong et al., 2024; Ghosh et al., 2025b; Goel et al., 2025) are built on pretrained LLM backbones with separate audio encoders. Their music data follows a recurring recipe: recordings, captions, metadata, and musical attributes extracted from audio provide source material, which language models turn into richer captions and QA pairs. AF3 compiles and releases AudioSkills-XL using this process, then combines it with other caption and QA datasets for training.

In follow-up work, Music Flamingo (Ghosh et al., 2025a) fine-tuned AF3 while expanding this data recipe, creating MF-Skills from YouTube recordings and publicly releasing its annotations and YouTube IDs. Audio Flamingo Next (Ghosh et al., 2026) also included MF-Skills in its music training mix. More recent models use related approaches: Nemotron 3 Nano Omni 30B-A3B (NVIDIA Team, 2026) uses a similar synthetic-data recipe for music QA, while Audex (Kong et al., 2026) reports following AF3 and Audio Flamingo Next in its audio-understanding data.

There has also been work on adapting pretrained LLMs for symbolic music understanding. ChatMusician (Yuan et al., 2024) continues training on MusicPile, which combines ABC-notation with music-related text, including summaries generated from YouTube track metadata. MIDI-LLaMA (Yang et al., 2026) adds a MIDI encoder and instruction-tunes it on QA and captions derived from GiantMIDI-Piano (Kong et al., 2022). Unlike MF-Skills and other caption/QA-based audio recipes above, we use the original sequence of spoken commentary and piano demonstrations from instructional recordings as text and MIDI, then test continued pretraining on these interleaved sequences.

3 PianoTalk: Interleaved Piano and Commentary

In this section, we overview our proposed data recipe behind PianoTalk. Specifically, we collect interleaved commentary and music transcripts at training scale (20k hours) from publicly-available piano instruction videos. Upon publication, we will publicly release the dataset as speech and MIDI transcripts, excluding the source audio.

3.1 Collecting and Processing Instructional Piano Videos

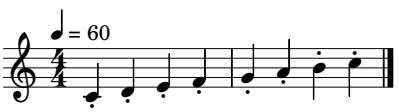
The objectives of our dataset pipeline are twofold. First, we aim to accurately identify videos of interleaved speech and solo-piano content at scale. Second, we aim to convert the videos into time-aligned commentary text and solo-piano MIDI transcriptions. Figure 4 in Appendix A.1 provides a schematic overview of the complete pipeline as a flowchart.

Source discovery. We use a two-stage LLM-assisted crawling procedure to discover candidate recordings. Starting from a small set of manually selected YouTube videos, the first stage evaluates video titles and descriptions for evidence of piano content, and iteratively expands the search by crawling related videos. This candidate pool seeds a second, more targeted discovery stage that aims to identify videos whose metadata suggests that they contain interleaved speech and piano. In addition to related videos, we crawl channels when at least five of their videos pass the speech-piano metadata filter or more than 25% of their scored videos pass the piano-content filter. We used `Gemma-4-26B-A4B` (Gemma Team et al., 2026) for metadata filtering in this stage. The scoring and expansion rules, together with the speech-piano selection prompt, are provided in Appendices A.1 and A.2.

Content detection. We discard LLM-selected videos whose auto-generated captions contain fewer than 100 words, then download only the audio from the remaining videos. To ensure relevant content, we use the same piano classification and transcription models as used in Bradshaw & Colton (2025). The classifier produces a per-second score trained as a proxy for the wet-dry mix of piano and other audio. We retain recordings with a mean score of at least 0.3 (manually tuned), isolate the piano with MVSEP[1] (Solovyev et al., 2024), and score the resulting piano stem with the same classifier. For recordings that pass this second filter, we transcribe the stem to MIDI with the transcription model and generate speech transcripts with Parakeet-TDT-0.6B-v3 (Sekoyan et al., 2025) to refine the auto-generated captions.

3.2 MIDI-Text Tokenization

We encode the raw transcribed MIDI into a text representation, which we refer to as *MIDI-text*. We designed this encoding to be token efficient, information dense, and parseable by existing LLM text tokenizers. We first tokenize each MIDI segment with the Aria tokenizer (Bradshaw et al., 2025), which quantizes velocity to 13 levels and represents onsets and durations at 10 ms resolution. We then format each note as pitch, velocity, onset, and duration, with a marker every 5 seconds that advances the time reference. Onsets are relative to the most recent marker, keeping their values bounded while preserving timing across the segment. The text header specifies the time units and note fields. For example, a C-major scale from C4 to C5 with notes 1 second apart and staccato durations (e.g., 150 milliseconds) is represented as follows (colors indicate individual tokens):



→

```
<piano>unit=10ms;advance=5s;
notes(midi_pitch,velocity,onset,duration):
60,90,0,15;62,90,100,15;64,90,200,15;
65,90,300,15;67,90,400,15;advance;
69,90,0,15;71,90,100,15;72,90,200,15;</piano>
```

3.3 Dataset Composition and Splits

In total, PianoTalk contains 138,212 videos. The speech intervals span 17,706 hours and the piano-note intervals span 14,931 hours, including 6,571 hours in which speech and piano overlap. At least one modality is present for 26,066 hours. The captions cover 44 languages, English (68.48%), Spanish (8.41%), and French (3.91%) being the most common caption languages. In Appendix A.3 we analyze speech-piano interleaving across the corpus. We provide canonical train, validation, and test splits containing 110,554, 13,837, and 13,821 videos, respectively (approximately 80/10/10). All videos from a given channel are assigned to the same split.

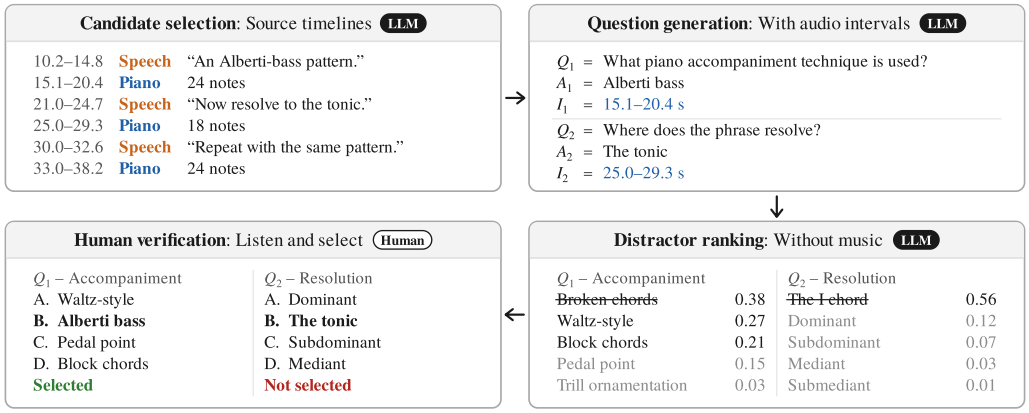


Figure 2: Construction of PianoBench (clockwise from top left). Speech and piano activity form a source timeline. An LLM proposes questions, answers, audio intervals, and distractors. A model ranks distractors without hearing the music. A human checks the passages and answer choices. Q1 is retained, but Q2 is rejected because too few suitable distractors remain.

4 PianoBench: Musical Perception and Reasoning

The instructional videos used to construct PianoTalk provide examples of solo-piano music with adjacent commentary, but proximity in the raw data alone does not make those examples ready for evaluating musical capabilities. In this section, we further process the piano stems, captions, and MIDI transcriptions from the PianoTalk test split to construct PianoBench, a set of 250 human-verified MCQs about short piano passages from 162 videos. Each question is evaluated separately with the same passage in three forms: source-separated piano audio, audio rendered from its MIDI transcription using a virtual piano (Pianoteq[2]), and music transcripts. We use no-music scoring to select distractors. In evaluation, a no-music control measures what models infer from the question and choices alone, allowing us to assess the contribution of each musical representation. Our construction pipeline is shown in Figure 2.

Candidate Selection. Using the PianoTalk test split, we first process the raw MIDI and speech captions into a structured list of contiguous chunks using the word and note timestamps. We define a chunk as any sequence of events with no more than 1 second between consecutive notes or 2 seconds between consecutive words. We format this information into a compressed representation containing chunk-start and end timestamps, speech-text, and absolute MIDI-note counts. We concatenate these representations to produce a coarse, interleaved content timeline for each video. To reduce our pool of candidate videos to a subset with content relevant for downstream question extraction, we use a lightweight LLM, Gemma-4-26B-A4B, to rank videos in our corpus using these timelines. An example timeline and the ranking prompt are given in Appendix B.1.

Question Generation. In order of priority determined by our ranking, we then use a more powerful LLM, GPT 5.6 Sol, to process timelines into question-answer pairs with corresponding piano-stem segment timestamps (up to five per video; see Appendix B.2). As the language model only has access to coarse indicators of piano activity, it is forced to rely on the surrounding commentary when suggesting questions. We follow a procedure inspired by Zang et al. (2025) to generate distractors. First, we use GPT 5.6 Terra to generate ten candidate distractors, removing semantically redundant, potentially correct, and ambiguous choices. For each candidate, we construct three four-way choice sets containing the correct answer, that candidate, and two randomly sampled candidates from the remaining pool. Without access to the audio or MIDI, we record the probability that Gemma-4-12B assigns to selecting the candidate as the correct answer in each choice set. We define the candidate's adversarial score as the mean of this probability across the three sets. The prompts for generation and scoring are given in Appendix B.3.

[1] https://github.com/zfturbo/music-source-separation-training/releases#release-v1.0.21

[2] We use Pianoteq 8 Stage with the NY Steinway D Classical Recording preset.

Human Verification. To ensure accuracy and improve the diversity of the question-answer pairs, we finally manually inspect and verify the candidate questions[3], listening to both the piano stem and MIDI rendering. We aimed to select a diverse set of questions with one unambiguous answer under each input representation, rejecting candidate questions that are incorrect, unverifiable, or of low quality, and preferring higher-scoring distractors while removing duplicate, irrelevant, or misleading choices missed by the distractor generator. A released example is shown in Appendix B.4.

The final music clips are 2–20 seconds long, with a median of 11 seconds. During human verification, we favored questions with clear piano stems and reliable MIDI transcriptions. We nevertheless identified a few faulty transcriptions after benchmarking models. These are tagged in the released dataset, but are included in the results reported in Section 5.2. We discuss their effect on evaluation in Section 5.3.

5 Diagnosing Musical Understanding in Current Models

We evaluate audio-language models and text-only language models on PianoBench across the three input conditions: source audio, Pianoteq rendering, and MIDI-text, as well as on a no-music control. Accuracy improvements over the no-music control (notated as Δ in this section) measure additional information that the models obtain from each musical input, while the difference between MIDI-text and audio compares reasoning over explicit note events with the ability to recover and reason about those events from audio. Comparing the source and Pianoteq conditions further tests whether audio performance is consistent across the original and a controlled rendering.

5.1 Experimental Setup

We benchmark three broad model classes: audio-language models, specialist audio and music-understanding models, and large language models with COT reasoning capabilities. We include families with publicly documented training recipes, including Nemotron/Audex and Audio/Music Flamingo, whose training mixtures include music captions, question-answer pairs, and automatic speech recognition data. We used model-specific inference configurations, including author-recommended settings where available. For configurations without a thinking mode, we generally used greedy decoding for MCQ answering. We report the exact hyperparameters and inference route for each model in Appendix C.1. Answer choices are shuffled once per question and held fixed across models and input conditions. As a basic check of the audio-input inference, we confirmed that each audio model identified several held-out piano clips as piano music. We request responses in JSON format and extract the answer, but also accept an isolated valid answer letter as a backup. If neither format can be parsed, we resubmit the prompt for up to five attempts in total. We report accuracy under each input condition and the percentage-point change (Δ) from the corresponding no-music accuracy. The prompts for all input conditions are provided in Appendix C.2. We evaluate all models zero-shot, without demonstrations or task-specific adaptation.

5.2 Results

Audio input. Table 1 reports accuracy across input formats. The benefit of audio differs substantially across model categories. For the open-weight audio-language models, changes relative to the no-music control are modest and varied, ranging from -1.6 to +6.0 percentage points across the two audio modalities where control comparisons are available. Audio Flamingo and Music Flamingo, which represent specialist audio-understanding models, obtain piano-stem accuracies of 37.2 and 42.4%, respectively. By contrast, the proprietary audio-language models improve across the board. Gemini 3.1 Pro achieves the highest audio accuracy of 80.4% (+16.8 points) when provided with the source audio. Qwen 3.5 Omni Plus has the next-largest gain over the control, at +11.2 points. Accuracies for the Pianoteq and source-audio conditions are broadly similar throughout.

MIDI-text input. Compared with audio, performance with MIDI-text is generally higher among models that demonstrate a clear benefit from musical input. Eleven audio-language models improve by at least five percentage points over their no-music control in one or more musical-input conditions. Of these, eight achieve their highest accuracy with MIDI-text; the exceptions are Qwen 3

[3]The human annotator was an author with prior musical training.

Model	No music	MIDI-text		Piano synth		Piano stem	
	Acc.	Acc.	Δ	Acc.	Δ	Acc.	Δ
Open-weight audio-language models							
Qwen 3 Omni 30B-A3B (thinking)	42.4	47.6	+5.2	48.4	+6.0	46.8	+4.4
Qwen 3 Omni 30B-A3B (non-thinking)	44.8	47.2	+2.4	50.8	+6.0	49.2	+4.4
Gemma 4 12B (thinking)	44.8	58.4	+13.6	46.4	+1.6	45.2	+0.4
Gemma 4 12B (non-thinking)*	-	-	-	51.2	-	51.6	-
Nemotron 3 Nano Omni 30B-A3B	46.4	45.6	−0.8	47.2	+0.8	48.4	+2.0
Nemotron Audex 30B-A3B	41.2	48.8	+7.6	42.8	+1.6	39.6	−1.6
Audio Flamingo 3†	-	-	-	38.8	-	37.2	-
Music Flamingo†	-	-	-	45.6	-	42.4	-
$\text{Inkling}_{\text{small}}$ (medium)	51.6	68.8	+17.2	52.8	+1.2	50.4	−1.2
Inkling (medium)	47.2	66.4	+19.2	47.6	+0.4	46.8	−0.4
Proprietary audio-language models							
Gemini 3.1 Pro Preview (medium)	63.6	89.2	+25.6	78.4	+14.8	80.4	+16.8
Qwen 3.5 Omni Plus	52.0	57.2	+5.2	62.4	+10.4	63.2	+11.2
Muse Spark 1.2 (medium)	48.8	86.8	+38.0	56.8	+8.0	55.6	+6.8
Gemini 3.7 Flash (medium)	62.0	90.0	+28.0	63.6	+1.6	66.8	+4.8
Gemini 3.8 Flash (medium)	62.0	88.8	+26.8	66.4	+4.4	64.0	+2.0
Language models evaluated on MIDI only							
Qwen 3.5 9B (thinking)	43.6	70.0	+26.4	-	-	-	-
Qwen 3.5 9B (non-thinking)	46.4	50.0	+3.6	-	-	-	-
GPT 5.6 Terra (high)	47.2	85.2	+38.0	-	-	-	-
GPT 5.6 Sol (none)	52.4	70.4	+18.0	-	-	-	-
GPT 5.6 Sol (medium)	56.0	85.6	+29.6	-	-	-	-
GPT 5.6 Sol (high)	56.4	87.2	+30.8	-	-	-	-
GPT 5.6 Sol (xhigh)	58.8	88.0	+29.2	-	-	-	-
GPT 6 Astra (high)	56.8	93.2	+36.4	-	-	-	-

Table 1: Zero-shot accuracy (%) on the 250 PianoBench MCQs. Models are evaluated with the question and choices alone (no music), MIDI-text, Pianoteq-synthesized audio, and the source-separated piano stem, where supported. Δ denotes the percentage-point change from that model's no-music accuracy. †: Audio Flamingo 3 and Music Flamingo had inconsistent instruction following on text-only inputs. ∗: Gemma 4 12B continued to produce chain-of-thought on MIDI-text and no-music inputs despite thinking being disabled; we therefore omit these results and report Δ over the thinking settings.

Omni Instruct/Thinking and Qwen 3.5 Omni Plus. The highest benchmark accuracies and largest improvements over the no-music control are achieved by strong reasoning models. GPT 6 Astra with high reasoning effort achieves the highest overall accuracy, at 93.2%.

No-music control. Accuracy on the no-music control varies from 41.2% (Nemotron Audex) to 63.6% (Gemini 3.1 Pro), well above the 25% chance level, showing that different models exploit language and answer-option priors to different degrees. This also highlights that, despite our attempts to generate plausible and robust distractors, the benchmark retains cues, possibly introduced by human distractor selection, that models can exploit without access to the music. The highest control accuracies are obtained by models with strong general reasoning and multimodal capabilities, such as the Gemini 3 family. We note that, although GPT 5.6 Terra and Gemma 4 12B were used to generate and rank distractors, respectively, their control accuracies remain within the range of other models in their respective categories. Given the variation in control accuracy, we primarily rely on within-model improvements over this control to assess the additional benefit provided by each musical input.

5.3 Discussion

Taken together, the results show a representation-dependent gap in access to musical information. The open-weight audio-language models obtain at most modest benefits from either audio representation where control comparisons are available. Audex's limited audio performance is no-

table because its documented musical training mixture largely comprises music captions and audio question-answering data. The stronger audio gains of Gemini 3.1 Pro and Qwen 3.5 Omni show that substantial performance improvements from audio are possible; however, little information about their data mixtures is public. This may suggest that simply scaling existing open-data recipes might not necessarily transfer to capabilities tested by PianoBench. For several models, the stronger 0-shot MIDI-text results show that at least part of the required musical reasoning becomes available when note events are made explicit. Their weaker audio results are consistent with a bottleneck in comparable musical information being available from sound. The text-only models Qwen 3.5 9B and GPT 5.6 Sol show large improvements with COT, suggesting that it helps them connect note events to higher-level musical concepts.

We also investigated the questions that GPT 6 Astra answered incorrectly and found that five involved transcription errors in questions that were missed by our human-verification pipeline[4]. The remaining errors showed a qualitative tendency to involve higher-level musical ideas expressed across longer passages containing many notes, e.g., Q: "What type of meter or dance-like accompaniment underpins this passage?"; A: "A waltz-like accompaniment in triple meter, with a triplet feel." We characterize these errors in Appendix C.3. While Astra's overall performance provides evidence that frontier reasoning capabilities generalize to 0-shot musical questions, this analysis also suggests that additional techniques may be needed to extend these capabilities to higher-level musical tasks.

6 Learning from Interleaved Musical Instruction

PianoBench reveals that explicit note events support strong performance in reasoning models, while smaller open-weight models without these capabilities make less use of MIDI-text. In this section, we ask whether PianoTalk can be used as training data to help models learn to reason over this representation. One approach would be to convert this data into automatically generated supervised descriptions or question-answer pairs like the aforementioned music-understanding data recipes. We instead adopt a continued-pretraining approach, using speech text and our MIDI-text representation as self-supervised training data. Specifically, we test whether placing temporally aligned speech and MIDI chunks within interleaved documents improves downstream reasoning over MIDI-text after instruction tuning, compared with training on the same blocks as separate speech and MIDI documents. Focusing on MIDI-text allows us to isolate the effect of interleaving while keeping the model architecture and training objective fixed, without introducing an audio encoder or adapting pretrained audio models and their training recipes to support this form of interleaving.

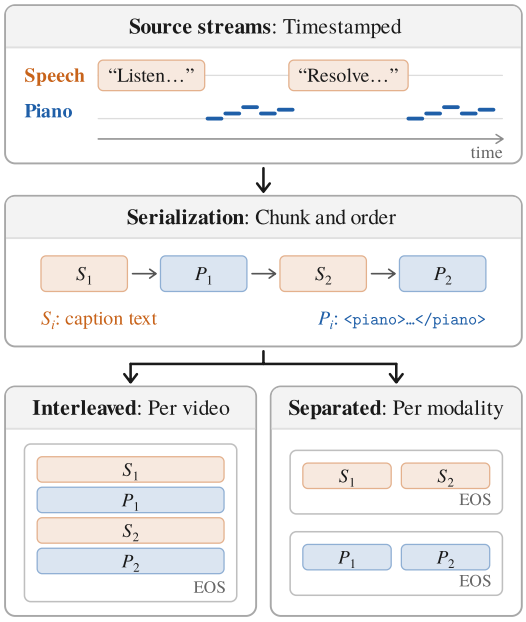


Figure 3: Construction of mid-training documents. Interleaved and separated conditions use the same speech and piano chunks, organized either chronologically in one interleaved document or as separate speech-only and piano-only documents.

6.1 Mid-Training and Instruction Tuning

Concretely, we mid-train (i.e., continue pretraining) `Qwen3.5-9B-Base` on our train dataset split under three conditions: speech only, separated speech and MIDI-text, and interleaved speech and MIDI-text. The speech-only condition controls for gains from instructional speech alone.

[4]These questions remain included in the reported accuracy.

To produce interleaved speech and MIDI-text documents, we split speech and MIDI streams at gaps longer than 2 seconds and add boundaries within longer passages at every 5 seconds. We then combine the chunks in chronological order, merging consecutive chunks of the same modality. This preserves continuous single-modality passages while producing finer-grained interleaving where speech and piano overlap. We concatenate the chunks as newline-separated text. We form the separated documents by splitting each final interleaved document into speech-only and MIDI-only lines, without additional segmentation or merging. Both conditions therefore use the same chunks. Figure 3 visualizes this process; the full construction and packing procedure is given in Appendix D.

Finally, we apply the same standard instruction-tuning phase to each checkpoint, finetuning on a mixture of general instruction-following data (40k sampled from the Tulu 3 SFT Mixture (Lambert et al., 2025)) and four-way multiple-choice questions (10k sampled from MMLU `auxiliary_train` split (Hendrycks et al., 2021)), reformatted to the response format of PianoBench. We chose Qwen 3.5 9B because it scores only 50.0% with MIDI-text without COT, despite reaching 70.0% with thinking enabled. We also retain the original base checkpoint as an SFT-only baseline. During instruction tuning we include neither MIDI-text question-answering data nor chain-of-thought supervision, allowing us to test whether a smaller model can instead acquire these capabilities through mid-training and transfer them zero-shot to PianoBench.

6.2 Experiments

Experimental Setup. We trained each model using the AdamW optimizer for one epoch over the corresponding document sets. We used a learning rate of $2e$-5 with a 5% warmup, followed by cosine decay to 10%. We used a sequence length of 4,096 tokens with an effective batch size of 2,621,440 tokens per gradient update. In total the speech-only, separated, and interleaved training corpora contained 169 million, 2,988 million, and 2,987 million tokens, respectively. In the instruction-tuning stage, we used the same learning rate schedule, and trained on 50,000 examples with an effective batch size of 128 examples per update.

Condition	**No music**	**MIDI-text**	
	Acc.	Acc.	Δ
Official Instruct	46.4	50.0	+3.6
Base-SFT	40.8	42.8	+2.0
Speech only	37.2	37.6	+0.4
Separated	42.4	42.4	+0.0
Interleaved	45.6	**56.0**	**+10.4**

Table 2: PianoBench accuracy (%) after mid-training and instruction tuning of Qwen 3.5 9B. Δ is the gain over no music (pp). Official Instruct is `Qwen/Qwen3.5-9B`[5]; Base-SFT is its base variant after our SFT. All responses use greedy decoding without chain of thought.

Results. Table 2 summarizes performance after mid-training and instruction tuning. Interleaved training achieves 56.0% accuracy with MIDI-text, a 10.4 percentage-point improvement over its no-music control, compared with gains of 0.0 points for separated training and 0.4 points for speech-only training. This provides evidence that preserving the temporal relationship between instructional speech and musical events improves downstream use of MIDI-text beyond exposure to the same content in separate documents.

7 Conclusion

In this paper, we have introduced PianoTalk and PianoBench, a new dataset and benchmark for evaluating and learning musical understanding in language models across audio and symbolic input formats. By evaluating audio-language models on PianoBench, we reveal a disparity between musical perception from audio and the text-based COT reasoning capabilities of language models when note events are made explicit. Through further mid-training experiments, we provide evidence that training on PianoTalk can improve reasoning *over* symbolic formats of music, such as MIDI-text, without COT. Future work could investigate whether combining these reasoning abilities with interleaved audio and text training helps narrow the gap between audio and symbolic input.

[5] https://huggingface.co/Qwen/Qwen3.5-9B

AI USE STATEMENT

In this work, we use generative AI tools for writing and editing code used in our experiments, and for generating and iterating on figures in the paper. We have also used LLMs in our dataset and benchmark construction pipelines (as described in Section 3.1 and Section 4). We used grammar checkers to assist with paper writing. We have not used generative AI tools to design or provide feedback on research methodology or experiments, or to interpret results. The text of the paper was also fully written by the authors. We have reviewed all AI-assisted work. All code generated by LLMs was verified and tested for correctness by the authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

Our work sits within the broader landscape of music and AI, and while there are some ethical considerations worth noting, we believe that the associated ethical risks are low. Our dataset is built from publicly available internet videos, which is consistent with common practice in large-scale data collection efforts in the field. We do not release the audio from these videos. We instead release a transformed version of the audio, the text and MIDI data we extract. Our focus is on evaluating and improving music understanding as a step towards supporting musicians in their creative process, as opposed to training generative models on copyrighted content, which raises far more significant concerns regarding the replacement of human creativity and labor. We also note that while our dataset covers 44 languages, it predominantly features widely spoken Western languages. This may introduce some degree of geographic bias, but is a byproduct of online content availability rather than a deliberate design choice.

REFERENCES

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob Menick, Sebastian Borgeaud, Andrew Brock, Aida Nematzadeh, Sahand Sharifzadeh, Mikolaj Binkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karen Simonyan. Flamingo: a visual language model for few-shot learning, 2022. URL https://arxiv.org/abs/2204.14198.

Emmanouil Benetos, Simon Dixon, Zhiyao Duan, and Sebastian Ewert. Automatic music transcription: An overview. *IEEE Signal Processing Magazine*, 36(1):20–30, 2018.

Louis Bradshaw and Simon Colton. Aria-MIDI: A dataset of piano MIDI files for symbolic music modeling, 2025. URL https://arxiv.org/abs/2504.15071.

Louis Bradshaw, Honglu Fan, Alexander Spangher, Stella Biderman, and Simon Colton. Scaling self-supervised representation learning for symbolic piano performance, 2025. URL https://arxiv.org/abs/2506.23869.

Brandon James Carone, Iran R. Roman, and Pablo Ripollés. The MUSE benchmark: Probing music perception and auditory relational reasoning in audio LLMs, 2025. URL https://arxiv.org/abs/2510.19055.

Alexandre Défossez. Hybrid spectrogram and waveform source separation, 2022. URL https://arxiv.org/abs/2111.03600.

Gemma Team, Sherif El Abd, Vaibhav Aggarwal, Robin Algayres, Alek Andreev, Olivier Bachem, Ian Ballantyne, Cormac Brick, Victor Cărbune, Michelle Casbon, et al. Gemma 4 technical report, 2026. URL https://arxiv.org/abs/2607.02770.

Sreyan Ghosh, Arushi Goel, Lasha Koroshinadze, Sang gil Lee, Zhifeng Kong, Joao Felipe Santos, Ramani Duraiswami, Dinesh Manocha, Wei Ping, Mohammad Shoeybi, and Bryan Catanzaro. Music Flamingo: Scaling music understanding in audio language models, 2025a. URL https://arxiv.org/abs/2511.10289.

Sreyan Ghosh, Zhifeng Kong, Sonal Kumar, S Sakshi, Jaehyeon Kim, Wei Ping, Rafael Valle, Dinesh Manocha, and Bryan Catanzaro. Audio Flamingo 2: An audio-language model with long-audio understanding and expert reasoning abilities, 2025b. URL https://arxiv.org/abs/2503.03983.

Sreyan Ghosh, Arushi Goel, Kaousheik Jayakumar, Lasha Koroshinadze, Nishit Anand, Zhifeng Kong, Siddharth Gururani, Sang gil Lee, Jaehyeon Kim, Aya Aljafari, Chao-Han Huck Yang, Sungwon Kim, Ramani Duraiswami, Dinesh Manocha, Mohammad Shoeybi, Bryan Catanzaro, Ming-Yu Liu, and Wei Ping. Audio Flamingo Next: Next-generation open audio-language models for speech, sound, and music, 2026. URL https://arxiv.org/abs/2604.10905.

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. Audio Flamingo 3: Advancing audio intelligence with fully open large audio language models, 2025. URL https://arxiv.org/abs/2507.08128.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding, 2021. URL https://arxiv.org/abs/2009.03300.

Qiuqiang Kong, Bochen Li, Jitong Chen, and Yuxuan Wang. GiantMIDI-Piano: A large-scale MIDI dataset for classical piano music, 2022. URL https://arxiv.org/abs/2010.07061.

Zhifeng Kong, Arushi Goel, Rohan Badlani, Wei Ping, Rafael Valle, and Bryan Catanzaro. Audio Flamingo: A novel audio language model with few-shot learning and dialogue abilities, 2024. URL https://arxiv.org/abs/2402.01831.

Zhifeng Kong, Sang gil Lee, Jaehyeon Kim, Boxin Wang, Zihan Liu, Sungwon Kim, Yang Chen, Arushi Goel, Rajarshi Roy, Wenliang Dai, Zhuolin Yang, Yangyi Chen, Dongfu Jiang, Sreyan Ghosh, Tuomas Rintamaki, Andrew Tao, Jonathan Raiman, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. Unified audio intelligence without regressing on text intelligence, 2026. URL https://arxiv.org/abs/2607.05196.

Sonal Kumar, Šimon Sedláček, Vaibhavi Lokegaonkar, Fernando López, Wenyi Yu, Nishit Anand, Hyeonggon Ryu, Lichang Chen, Maxim Plička, Miroslav Hlaváček, William Fineas Ellingwood, Sathvik Udupa, Siyuan Hou, Allison Ferner, Sara Barahona, Cecilia Bolaños, Satish Rahi, Laura Herrera-Alarcón, Satvik Dixit, Siddhi Patil, Soham Deshmukh, Lasha Koroshinadze, Yao Liu, Leibny Paola Garcia Perera, Eleni Zanou, Themos Stafylakis, Joon Son Chung, David Harwath, Chao Zhang, Dinesh Manocha, Alicia Lozano-Diez, Santosh Kesiraju, Sreyan Ghosh, and Ramani Duraiswami. MMAU-Pro: A challenging and comprehensive benchmark for holistic evaluation of audio general intelligence, 2025. URL https://arxiv.org/abs/2508.13992.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Chris Wilhelm, Luca Soldaini, Noah A. Smith, Yizhong Wang, Pradeep Dasigi, and Hannaneh Hajishirzi. Tulu 3: Pushing frontiers in open language model post-training, 2025. URL https://arxiv.org/abs/2411.15124.

Ziyang Ma, Yinghao Ma, Yanqiao Zhu, Chen Yang, Yi-Wen Chao, Ruiyang Xu, Wenxi Chen, Yuanzhe Chen, Zhuo Chen, Jian Cong, Kai Li, Keliang Li, Siyou Li, Xinfeng Li, Xiquan Li, Zheng Lian, Yuzhe Liang, Minghao Liu, Zhikang Niu, Tianrui Wang, Yuping Wang, Yuxuan Wang, Yihao Wu, Guanrou Yang, Jianwei Yu, Ruibin Yuan, Zhisheng Zheng, Ziya Zhou, Haina Zhu, Wei Xue, Emmanouil Benetos, Kai Yu, Eng-Siong Chng, and Xie Chen. MMAR: A challenging benchmark for deep reasoning in speech, audio, music, and their mix, 2025. URL https://arxiv.org/abs/2505.13032.

NVIDIA Team. Nemotron 3 Nano Omni: Efficient and open multimodal intelligence, 2026. URL https://arxiv.org/abs/2604.24954.

S Sakshi, Utkarsh Tyagi, Sonal Kumar, Ashish Seth, Ramaneswaran Selvakumar, Oriol Nieto, Ramani Duraiswami, Sreyan Ghosh, and Dinesh Manocha. MMAU: A massive multi-task audio understanding and reasoning benchmark, 2024. URL https://arxiv.org/abs/2410.19168.

Monica Sekoyan, Nithin Rao Koluguri, Nune Tadevosyan, Piotr Zelasko, Travis Bartley, Nikolay Karpov, Jagadeesh Balam, and Boris Ginsburg. Canary-1B-v2 & Parakeet-TDT-0.6B-v3: Efficient and high-performance models for multilingual ASR and AST, 2025. URL https://arxiv.org/abs/2509.14128.

Roman Solovyev, Alexander Stempkovskiy, and Tatiana Habruseva. Benchmarks and leaderboards for sound demixing tasks, 2024. URL https://arxiv.org/abs/2305.07489.

Tomáš Sourada, Katia Vendrame, and Jan Hajič, Jr. Music I care about: Automated multimodal benchmarking of LLM music perception skills on (almost) any music, 2026. URL https://arxiv.org/abs/2607.06015.

Benno Weck, Ilaria Manco, Emmanouil Benetos, Elio Quinton, George Fazekas, and Dmitry Bogdanov. MuChoMusic: Evaluating music understanding in multimodal audio-language models, 2024. URL https://arxiv.org/abs/2408.01337.

Benno Weck, Pablo Puentes, Andrea Poltronieri, Satyajeet Prabhu, and Dmitry Bogdanov. HumMusQA: A human-written music understanding QA benchmark dataset. In *Proceedings of the 4th Workshop on NLP for Music and Audio (NLP4MusA 2026)*, pp. 58–67. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.nlp4musa-1.9. URL http://dx.doi.org/10.18653/v1/2026.nlp4musa-1.9.

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, Baosong Yang, Bin Zhang, Ziyang Ma, Xipin Wei, Shuai Bai, Keqin Chen, Xuejing Liu, Peng Wang, Mingkun Yang, Dayiheng Liu, Xingzhang Ren, Bo Zheng, Rui Men, Fan Zhou, Bowen Yu, Jianxin Yang, Le Yu, Jingren Zhou, and Junyang Lin. Qwen3-Omni technical report, 2025. URL https://arxiv.org/abs/2509.17765.

Meng Yang, Jon McCormack, Maria Teresa Llano, Wanchao Su, and Chao Lei. MIDI-LLaMA: An instruction-following multimodal LLM for symbolic music understanding, 2026. URL https://arxiv.org/abs/2601.21740.

Ruibin Yuan, Hanfeng Lin, Yi Wang, Zeyue Tian, Shangda Wu, Tianhao Shen, Ge Zhang, Yuhang Wu, Cong Liu, Ziya Zhou, Ziyang Ma, Liumeng Xue, Ziyu Wang, Qin Liu, Tianyu Zheng, Yizhi Li, Yinghao Ma, Yiming Liang, Xiaowei Chi, Ruibo Liu, Zili Wang, Pengfei Li, Jingcheng Wu, Chenghua Lin, Qifeng Liu, Tao Jiang, Wenhao Huang, Wenhu Chen, Emmanouil Benetos, Jie Fu, Gus Xia, Roger Dannenberg, Wei Xue, Shiyin Kang, and Yike Guo. ChatMusician: Understanding and generating music intrinsically with LLM, 2024. URL https://arxiv.org/abs/2402.16153.

Yongyi Zang, Sean O'Brien, Taylor Berg-Kirkpatrick, Julian McAuley, and Zachary Novack. Are you really listening? boosting perceptual awareness in Music-QA benchmarks, 2025. URL https://arxiv.org/abs/2504.00369.

A Video Discovery and Selection

A.1 Candidate discovery

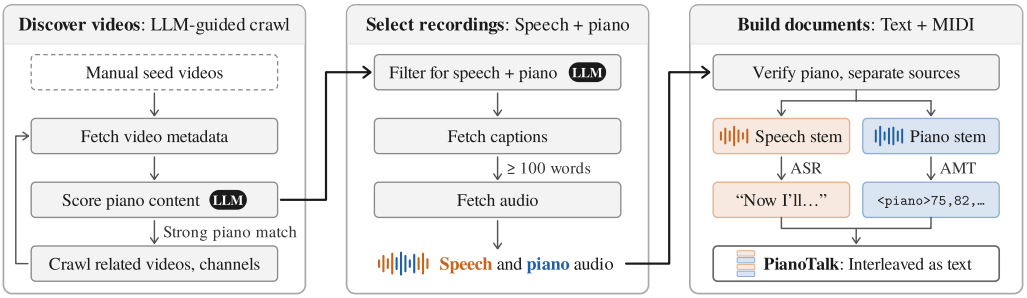


Figure 4: PianoTalk data recipe. An LLM-guided crawl expands from seed videos, using metadata to select videos likely to contain both speech and piano. Audio from videos with captions is fetched and screened for piano content. Audio is converted into interleaved speech and music transcripts via source separation, automatic speech recognition (ASR) and automatic music transcription (AMT).

We score video titles and the first 512 characters of their descriptions for likely piano content, beginning with manually selected seeds. The first prompt assigns scores from 0 to 6. Scores 0, 1, and 2 indicate no evidence of piano, incidental piano, and piano-adjacent content, respectively. Score 3 indicates piano-related content with uncertain solo-piano audio; scores 4 and 5 indicate likely and very likely contiguous piano-only audio. Score 6 additionally requires evidence of uncommon repertoire, improvisation, or a distinctive arrangement. Videos scoring at least 5 seed searches through related videos; videos scoring 3 are evaluated by the speech-piano prompt below.

We expand a channel when at least five of its videos pass the speech-piano metadata filter, or when more than 25% of its scored videos have piano-content scores of at least 5. Videos found through either route enter the same scoring process.

A.2 Speech–piano selection

The second prompt scores whether a video contains spoken commentary and related solo-piano examples. It assigns 0 to unlikely matches, 4 to possible matches, 5 to likely matches, and 6 to clear matches. Scores of at least 4 pass the metadata filter. The prompt is reproduced in full below.

```
Analyze the YouTube video title and description.

Task:
Score how likely the video is to contain useful paired spoken language and
solo-piano audio.

The intended dataset uses timed captions together with nearby piano examples.
Speech and piano may alternate or overlap. Useful videos should likely contain:

1. Meaningful spoken instruction, explanation, analysis, or commentary
2. Audible solo-piano examples in the same video
3. A meaningful relationship between the speech and piano examples

The video was selected from piano-related metadata, but piano playing is not
guaranteed. Judge only from the supplied title and description.

Good candidates include spoken piano tutorials or lessons, harmony/theory
breakdowns, masterclasses, technique explanations, composition or performance
analysis at the piano, and narrated piano or virtual-piano demonstrations.

Use only these scores:

6 = Clear high-yield match.
The metadata explicitly indicates substantial spoken explanation together with
related solo-piano demonstrations, passages, or examples.

5 = Likely useful candidate.
```

```
The metadata suggests meaningful speech and related solo-piano examples in the
same video, but does not give enough concrete detail for 6.

4 = Possible candidate.
The metadata is sparse or ambiguous, but related speech and solo-piano examples
are plausible. This includes ordinary piano tutorials or lessons unless the
metadata clearly indicates a non-verbal format.

0 = Unlikely or wrong format.
Use 0 for performance-only videos, speech without related piano examples,
unrelated background piano, vocals, ensemble, or clearly non-verbal piano
content.

Do not infer spoken narration merely because the written description explains
the music. Interviews, reactions, reviews, and discussions score 5 or 4 only
when the metadata also suggests related solo-piano examples in the same video.

If both title and description are missing or blank, use 0.

Return only JSON with one of the scores 6, 5, 4, or 0. Do not use a markdown
fence:
{"score": N}
```

A.3 Measuring interleaving

During dataset construction, recordings were selected if they had piano and speech content independently. To quantify the degree of interleaving, we split speech and MIDI independently at pauses longer than 2 seconds, then add candidate boundaries within longer passages using a 5-second window. We sort these candidate chunks by start time and count adjacent speech-MIDI switches. For $N > 2$ candidates and S switches, the score is $100(S-1)/(N-2)$. A recording consisting of a single speech segment followed by a single MIDI segment scores zero, while switching at every boundary scores 100. Figure 5 shows the distribution of these scores across the corpus.

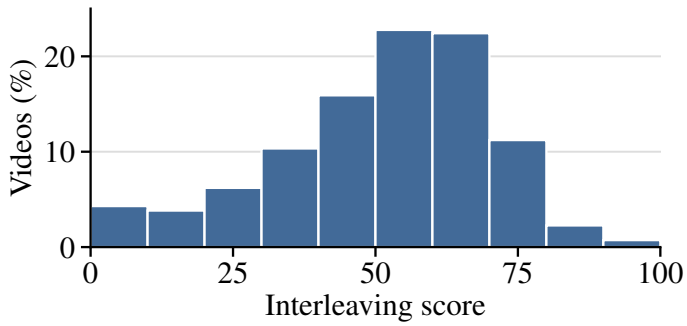


Figure 5: Distribution of interleaving scores across PianoTalk videos, measured from chronological speech and MIDI chunks. Most scores fall near the middle, indicating consistent alternation between commentary and piano.

B PianoBench Construction Details

B.1 Candidate Timelines and Video Ranking

The following timeline records begin in the first minute of the test-split video that produced released question 350. Speech about a tritone substitution alternates with five piano regions, including the two-note example beginning at 53.07 s. We reconstructed this excerpt from the released YouTube captions and full-length MIDI; the piano-stem silence check was unavailable in the release. Each JSON object is one record, with display-only wrapping of long lines.

```
{"type": "speech", "start_s": 0.12, "end_s": 4.92, "text": "this tune uses one of the biggest chord
↪  substitutions in"}
{"type": "speech", "start_s": 4.92, "end_s": 5.71, "text": "jazz"}
{"type": "piano", "start_s": 5.84, "end_s": 18.31, "number_of_notes_in_segment": 109}
{"type": "speech", "start_s": 18.08, "end_s": 22.97, "text": "that satin doll the Duke Ellington hit
↪  written by Billy strayhorn and the"}
```

```
{"type": "speech", "start_s": 22.98, "end_s": 26.48, "text": "classic substitution that it uses is called
↪  a tritone"}
{"type": "speech", "start_s": 26.48, "end_s": 31.37, "text": "substitution G7 has these two really really
↪  important"}
{"type": "piano", "start_s": 27.63, "end_s": 29.06, "number_of_notes_in_segment": 4}
{"type": "piano", "start_s": 30.23, "end_s": 31.73, "number_of_notes_in_segment": 10}
{"type": "speech", "start_s": 31.38, "end_s": 36.12, "text": "notes that are in them another chord that
↪  has those two really really important notes"}
{"type": "speech", "start_s": 36.12, "end_s": 40.92, "text": "is d flat seven it also has those"}
{"type": "piano", "start_s": 38.2, "end_s": 40.25, "number_of_notes_in_segment": 16}
{"type": "speech", "start_s": 40.92, "end_s": 45.36, "text": "notes so if both those chords have the
↪  important notes well you can just replace"}
{"type": "speech", "start_s": 45.36, "end_s": 50.28, "text": "one for another and that's what happens in
↪  satin doll so that's the substitution part well why is"}
{"type": "speech", "start_s": 50.28, "end_s": 55.199, "text": "it called a tritone substitution well G and
↪  D"}
{"type": "piano", "start_s": 53.07, "end_s": 58.09, "number_of_notes_in_segment": 2}
{"type": "speech", "start_s": 55.199, "end_s": 60.059, "text": "flat are a tritone apart so if satin all
↪  didn't have a tritone"}
```

Gemma-4-26B-A4B assigns each video an interest score from 1 to 6 based on its timeline. Higher-scoring videos are processed first for question generation. The complete ranking prompt is shown below. Angle-bracketed text denotes the runtime timeline substitution; all other text is reproduced verbatim.

```
Judge the strongest part of this timeline as a source for an audio-only piano
question. One strong part is enough; ignore video length and opportunity count.

The goal is a benchmark of musical understanding. The answer to an interesting
question should preserve a specific insight worth learning about the logic or
significance of the music, not merely the mechanics of producing it or an
obvious feature of the sound.

6 = Excellent material for an exceptionally revealing advanced benchmark question.
5 = Clear evidence of an advanced, substantive musical insight.
4 = Possibly advanced, with important uncertainty.
3 = Meaningful but intermediate or borderline musical understanding.
2 = Only routine, introductory, surface-level, or instrumental-mechanics material.
1 = No plausible meaningful material.

If the strongest material is only an instrumental technique or practice
pattern, it belongs at 2 or 3 even when it is difficult, systematic, or
organized using music theory.

Give 6 when the speech makes a clear, advanced insight available through the
associated piano music: the resulting question and answer could retain some of
the explanatory value of the source. The insight need not be novel to experts.
Judge substance, not impressive vocabulary. Assume speech and piano are
generally well paired. Do not propose questions.

Return only JSON:
{"score": N}

TIMELINE:
<timeline JSON objects, one per line>
```

B.2 Question Generation

GPT 5.6 Sol receives the timeline and proposes up to five questions per video, each with an integer-second clip interval no longer than 20 s, a fixed correct answer, and an evidence summary. It does not receive the piano audio or detailed MIDI notes. The speech text guides it toward candidate passages, but the eventual respondent receives only the isolated piano passage, question, and choices. The prompt below presents the question-generation instructions in API form: instructions about agent file and tool access are omitted, while the content criteria are retained. The runtime timeline substitution is marked in angle brackets.

```
Create benchmark-style questions about isolated piano audio.

All source information is included in this prompt. Work only from that
information.

Input:

TIMELINE: A chronological sequence of JSON objects, one per line, ordered by
each region's onset time.
```

```
- A speech region contains `type: "speech"`, `start_s`, `end_s`, and `text`.
- A piano region contains `type: "piano"`, `start_s`, `end_s`, and
  `number_of_notes_in_segment`.

Read the timeline in order and use the temporal relationship between speech and
nearby piano regions to locate piano passages that may support clear audio-only
questions. Speech is hidden metadata and will not be available to the listener.
Piano regions describe the source-separated, piano-only audio that will be
available to the listener.

Regions may overlap even though they are ordered by onset. Use their explicit
start and end times rather than assuming that one region ends when the next
begins. Continuous speech may be divided into consecutive timestamped regions
without a silence between them.

The detected note count confirms piano activity and may provide a rough clue
about musical density. It does not reveal the pitches, chords, rhythm,
articulation, technique, or musical identity of the passage.

For each QA pair, `audio_timestamp` is used to extract a standalone clip from
the source-separated piano audio. The listener receives only that clip, the
question, and the choices. Playback starts at 0 and the source timeline is not
available. Source timestamps are extraction metadata only and must not appear
in the question or choices. Refer to positions within the clip using terms such
as "at the beginning," "later," or "at the end."

Task:

Aim to generate 2 to 5 interesting and reliable questions about musical
content by turning concrete spoken observations about nearby piano passages
into distinctions that should be audible in a short isolated clip. Use
independent local events or passages where possible. Return fewer than two,
including zero, only when the supplied timeline genuinely lacks enough
concrete, speech-tied piano events to support them.

For each retained question, establish exactly one fixed correct answer, the
shortest practical integer-second clip, a concise timeline-supported evidence
summary, evidence provenance, and uncertainty. Include only supplied evidence;
do not imply that you heard the audio or invent musical details. Do not
generate, propose, or consider any incorrect answer choices yourself. They are
generated in a later stage.

Audio-grounding rules:

- Design each question so that a listener should be able to answer it from the
  selected piano clip without transcript, speech, visuals, or outside context.
- Do not refer to speakers, instructors, narration, transcripts, videos, lessons,
  or visual content.
- Do not ask what was said, intended, named, requested, or explained.
- Use speech to infer candidate questions about nearby piano passages. Because
  you cannot hear the audio, treat spoken descriptions as evidence rather than
  guaranteed truth.
- Favor concrete musical claims that speech ties to a specific passage and that
  are likely to be clearly audible in the isolated piano clip.
- The extracted clip must be no longer than 20 seconds and must contain all
  audio needed to answer the question.
- Choose the shortest practical audio range. Prefer a single local event or
  compact contrast; do not span distant passages or include long intervening
  audio.
- Do not infer specific musical details from note count alone.
- A comparison, change, or before-and-after question must include all passages
  needed for the comparison within the selected clip.
- Reject claims that require speech, notation, visuals, or facts not established
  by the audio, as well as musical details that would be too subtle to verify
  reliably. Recognition of conventional musical characteristics and categories
  from sound counts as audio understanding, not outside knowledge.

Question and correct-answer rules:

- Each question must have exactly one correct answer that is clearly supported
  by the audio.
- Avoid yes/no questions, subjective judgments, trivia, generic activity
  labels, and questions requiring perfect pitch unless the clip supplies an
  audible reference pitch.
- Consider the full range of musically meaningful information recoverable from
  the clip, from local audible details and relationships to broader
  characterization of the passage. Do not default to instructional details or
  before-and-after demonstrations merely because they are easier to locate.
- Fix the question and correct answer before distractor generation. Do not
  revise either one based on the distractor candidates.

Return JSON only, using this schema:
```

```
[
  {
    "id": 0,
    "audio_timestamp": [start_second, end_second],
    "question": "Question text",
    "correct_answer": "Correct answer",
    "evidence": "Concise description supported by the timeline",
    "provenance": ["Relevant supplied timeline observation"],
    "uncertainty": "low"
  }
]

IDs must be consecutive from zero. Both `audio_timestamp` values must be
nonnegative integer seconds from the start of the source, the end must be
greater than the start and no more than 20 seconds later. `provenance` must be a
nonempty list of concise strings, and `uncertainty` must be `low`, `medium`, or
`high`.

TIMELINE:
<timeline JSON objects, one per line>
```

B.3 Distractor Generation and Scoring

For each fixed question and answer, GPT 5.6 Terra proposes ten candidate distractors using the evidence summary from question generation. It removes choices it judges redundant, ambiguous, potentially correct, or irrelevant before scoring. The model has no access to the piano audio. The prompt below presents these instructions in API form, with runtime question data marked in angle brackets. The implementation supplied this data and separately scored the returned candidates.

```
Generate distractors for one fixed piano-audio question. Use only the complete
question data supplied below. You do not have the audio.

Generate `candidate_count` plausible incorrect answers. Each candidate must be
clearly wrong given the supplied textual audio evidence, while remaining
plausible to a model that sees only the question and answer choices. Match the
correct answer's musical dimension, grammar, specificity, length, and
terminology; prefer small counterfactual changes over conspicuous foils.

Before returning, remove candidates that are duplicated, semantically redundant,
ambiguous, potentially correct, off-axis, malformed, or unsupported by the
evidence. Keep at least three materially distinct candidates.

Return JSON only, using this shape:

{
  "candidates": ["Candidate 1", "Candidate 2", "Candidate 3"]
}

COMPLETE QUESTION DATA

{
  "question_id": <question_id>,
  "audio_timestamp": [<start_second>, <end_second>],
  "question": "<question>",
  "correct_answer": "<correct_answer>",
  "evidence": "<evidence>",
  "provenance": <provenance JSON array>,
  "uncertainty": "<uncertainty>",
  "candidate_count": 10
}
```

For each remaining candidate, the scoring tool constructs three four-choice sets containing the correct answer, that candidate, and two other sampled candidates. It shuffles the choices and asks Gemma-4-12B to answer using only the question and choices. The candidate score is the mean probability Gemma assigns to the candidate's choice letter across the three sets. This is a ranking of how tempting the incorrect answer appears without music; it is not a probability that the answer is musically correct. The text-only scoring prompt follows. The {`question`} placeholder receives the question text, and {`choices`} receives the four labelled answer choices.

```
Answer this music multiple-choice question using only its text. Choose the
option that appears most likely to be correct.

QUESTION:
{question}

CHOICES:
```

```
{choices}

Return only the single letter A, B, C, or D.
```

B.4 A Released Benchmark Example

After listening to the separated piano audio and the MIDI rendering, an author retained questions whose answer was unambiguous under both representations and rejected unsuitable distractors. The highest-scoring three remaining distractors were selected, and the four choices were shuffled once for the released benchmark. The following is the complete plain-text MIDI input sent to the evaluated model for question 1037, copied from the released PianoBench export. Long lines wrap only for display. The clip lasts six seconds.

```
Answer the multiple-choice question about the isolated piano performance.
Use only the compact piano note table below as evidence. Choose exactly one
option.

Table interpretation:
- `unit` is the duration of one timing unit; `advance` is the duration of one
  time window.
- Each `midi_pitch,velocity,onset,duration;` record describes one note; MIDI
  pitch 60 is middle C.
- `onset` is measured in timing units from the start of the current window.
- `duration` is measured in timing units and may extend beyond that window.
- Each `advance;` moves the current window forward by the header's `advance`
  duration.

PIANO NOTE TABLE:
<piano>unit=10ms;advance=5s;notes(midi_pitch,velocity,onset,duration):41,50,0,53;45,60,50,47;48,60,94,48;
↪  50,40,140,47;51,70,185,46;50,60,231,46;48,60,276,41;45,60,316,45;41,60,360,96;</piano>

QUESTION:
What kind of bass accompaniment pattern is played?

CHOICES:
A. An Alberti bass pattern.
B. A walking bass line.
C. A root-and-fifth bass pattern.
D. A pedal-point bass line.

Return only a JSON object with a single "choice" field whose value is one of
"A", "B", "C", or "D".
```

The correct choice is B, a walking bass line.

C Benchmark Evaluation Details

C.1 Model Inference Settings

Table 3 reports the requested inference settings for each configuration in Table 1. We asked all models to return a JSON object containing one answer letter. This specifies the final response format but does not disable a model's reasoning process. A dash in the sampling columns means that the parameter was omitted from the request. In the thinking column, it means that thinking was disabled or no separate thinking setting was available. Repeated model identifiers correspond to evaluations with different thinking settings.

Model	Inference route	Temp.	Top-p	Thinking
Self-hosted				
Qwen 3 Omni (thinking)	`huggingface:Qwen/Qwen3-Omni-30B-A3B-Thinking`	0.6	0.95	1024
Qwen 3 Omni (instruct)	`huggingface:Qwen/Qwen3-Omni-30B-A3B-Instruct`	0	1	–
Gemma 4 12B (thinking)	`huggingface:google/gemma-4-12B-it`	1	0.95	1024
Gemma 4 12B (non-thinking)	`huggingface:google/gemma-4-12B-it`	0	1	–
Nemotron Audex 30B-A3B	`huggingface:nvidia/Nemotron-Labs-Audex-30B-A3B`	0	1	–
Audio Flamingo 3	`huggingface:nvidia/audio-flamingo-3-hf`	0	1	–
Music Flamingo	`huggingface:nvidia/music-flamingo-hf`	0	1	–
Qwen 3.5 9B (thinking)	`huggingface:Qwen/Qwen3.5-9B`	1	0.95	16000
Qwen 3.5 9B (non-thinking)	`huggingface:Qwen/Qwen3.5-9B`	0	1	–
Provider-hosted				
Nemotron 3 Nano Omni	`openrouter:nvidia/nemotron-3-nano-omni-30b-a3b-reasoning:free`	0.6	0.95	Enabled
$\text{Inkling}_{\text{small}}$ (medium)	`openrouter:thinkingmachines/inkling-small`	1	1	Medium
Inkling (medium)	`openrouter:thinkingmachines/inkling`	1	1	Medium
Gemini 3.1 Pro (medium)	`openrouter:google/gemini-3.1-pro-preview`	–	–	Medium
Qwen 3.5 Omni Plus	`alibaba:qwen3.5-omni-plus-2026-03-15`	–	–	–
Muse Spark 1.2 (medium)	`openrouter:meta/muse-spark-1.2`	0	1	Medium
Gemini 3.7 Flash (medium)	`openrouter:google/gemini-3.7-flash`	–	–	Medium
Gemini 3.8 Flash (medium)	`openrouter:google/gemini-3.8-flash`	–	–	Medium
GPT 5.6 Terra (high)	`openai:gpt-5.6-terra`	–	–	High
GPT 5.6 Sol (none)	`openai:gpt-5.6-sol`	–	–	–
GPT 5.6 Sol (medium)	`openai:gpt-5.6-sol`	–	–	Medium
GPT 5.6 Sol (high)	`openai:gpt-5.6-sol`	–	–	High
GPT 5.6 Sol (xhigh)	`openai:gpt-5.6-sol`	–	–	Xhigh
GPT 6 Astra (high)	`openai:gpt-6-astra`	–	–	High

Table 3: Requested temperature, top-p, and thinking settings for the models evaluated on PianoBench. Model labels abbreviate those in Table 1 where the identity remains clear.

C.2 Evaluation Prompts

No-music control. The model receives the question and four answer choices without music. The prompt is shown below; {`question`} and {`choices`} are replaced at evaluation time.

```
Answer the multiple-choice question about an isolated piano performance.
No representation of the performance is provided; answer using only the
question and answer choices. Choose exactly one option.

QUESTION:
{question}

CHOICES:
{choices}

Return only a JSON object with a single "choice" field whose value is one of
"A", "B", "C", or "D".
```

MIDI-text. The model receives the same question and choices with the piano passage as a MIDI-text note table. The additional {midi_document} placeholder is replaced with the note table.

```
Answer the multiple-choice question about the isolated piano performance.
Use only the compact piano note table below as evidence. Choose exactly one
option.

Table interpretation:
- `unit` is the duration of one timing unit; `advance` is the duration of one
  time window.
- Each `midi_pitch,velocity,onset,duration;` record describes one note; MIDI
  pitch 60 is middle C.
- `onset` is measured in timing units from the start of the current window.
- `duration` is measured in timing units and may extend beyond that window.
- Each `advance;` moves the current window forward by the header's `advance`
  duration.

PIANO NOTE TABLE:
{midi_document}

QUESTION:
{question}

CHOICES:
{choices}

Return only a JSON object with a single "choice" field whose value is one of
"A", "B", "C", or "D".
```

Audio. The model receives the same question and choices with an audio attachment. We use this prompt for both the source-separated piano stem and the Pianoteq rendering.

```
Answer the multiple-choice question about the isolated piano performance.
Use only the attached audio as evidence. Choose exactly one option.

QUESTION:
{question}

CHOICES:
{choices}

Return only a JSON object with a single "choice" field whose value is one of
"A", "B", "C", or "D".
```

C.3 GPT 6 Astra Error Analysis

Table 4 lists the 17 questions GPT 6 Astra answered incorrectly with MIDI-text input. The #notes column gives the number of notes in the supplied transcription. The Error column is reserved for Yes/No annotations of transcription errors that affect the question.

ID	Question	#notes	Error
36	What rhythmic style characterizes the left-hand accompaniment?	255	No
99	What articulation best describes the notes in the short repeated figure?	59	No
128	What is the relationship between the low repeated notes and the melody?	113	No
138	What recurring pulse pattern supports the melody?	44	No
157	Where does the harmony move later in the clip relative to the opening chord?	8	Yes
182	How does the chord quality change from the first chord to the next?	3	Yes
393	How many times does the brief four-note figure recur in the clip?	3	Yes
564	What chord quality is heard in the passage?	9	Yes
614	What change in chord quality redirects the progression toward a new key?	24	No
665	Which tonal mode characterizes the passage?	108	No
676	What overall character is established at the beginning of this passage?	205	No
685	How many notes make up the chord heard in this clip?	2	Yes
730	What is the overall direction of the chord progression in this clip?	36	No
732	What harmonic-quality change occurs later in the clip?	57	No
733	What kind of meter and rhythmic feel characterizes this passage?	47	No
758	What type of meter or dance-like accompaniment underpins this passage?	91	No
1031	How many contrapuntal voices are sounding in the texture?	72	No

Table 4: Questions answered incorrectly by GPT 6 Astra with MIDI-text input. The 17 errors correspond to 93.2% accuracy on 250 questions.

D Mid-Training Document Construction

To produce interleaved speech and MIDI-text documents, we: **(1)** Independently split speech and MIDI streams at gaps longer than 2 seconds. **(2)** Within each resulting span, add a candidate boundary at an event onset once at least 5 seconds have elapsed since the previous boundary, provided at least 5 seconds of material remains. **(3)** Combine speech and MIDI chunks and sort them by start time. **(4)** Merge consecutive chunks of the same modality.

This preserves continuous single-modality passages while producing finer-grained interleaving where speech and piano overlap. We then produce the training documents by concatenating newline-separated chunks, wrapping MIDI-text in `<piano>` XML tags. We form the separated documents by splitting each final interleaved document into its speech and MIDI lines, without additional segmentation or merging. The speech-only condition uses only the resulting speech documents. Figure 3 visualizes this process. Finally, during training we shuffle documents, tokenize them with the model's tokenizer, append EOS, and pack the resulting stream into 4,096-token training sequences.