Continuous Audio Thinking for Large Audio Language Models

Gyojin Han[1*] **Dong-Jae Lee**[1*] **Changho Choi**[1*] **Jongsuk Kim**[2*] **Junmo Kim**[1]
[1]KAIST, South Korea [2]PYLER, South Korea
{hangj0820, jhtwosun, ccho4702, junmo.kim}@kaist.ac.kr, js.kim@pyler.tech

Abstract

Large audio language models (LALMs) have shown impressive capabilities on diverse audio understanding tasks, ranging from speech transcription to music analysis. However, because LALMs are typically trained to produce text-aligned responses, their hidden states are progressively shaped for text generation rather than for preserving acoustic information. As a result, the diverse acoustic content that audio carries, such as phonetic detail, prosody, sound events, affect, and pitch, is lost along the way and difficult to leverage in the response. We introduce Continuous Audio Thinking (CoAT), a framework that equips audio language models with a continuous latent workspace for organizing acoustic information prior to response generation, grounded by distillation from audio experts. Within the thinking space, the model can utilize the rich acoustic information provided by expert distillation when generating its response. Furthermore, the proposed continuous thinking block can be processed in a single prefill, so CoAT does not require additional autoregressive decoding cost over the baseline. Across three LALMs, Qwen2-Audio, Qwen2.5-Omni-7B, and Audio Flamingo 3, performance gains on a broad benchmark suite spanning audio reasoning, audio understanding, music classification, speech emotion, and speech transcription demonstrate the effectiveness of CoAT. Further analysis confirms that the auxiliary supervision propagates from the thinking positions to the model's textual responses.

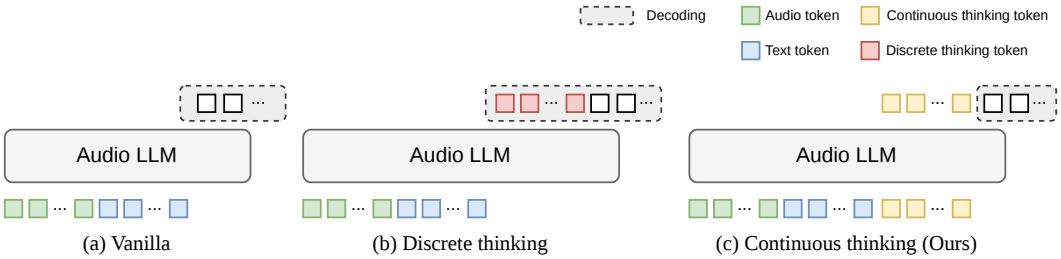
1 Introduction

Large Audio Language Models (LALMs) [9, 19, 49, 55] have established themselves as a natural interface for understanding speech, environmental sounds, music, and other acoustic signals through language. These models couple an audio encoder with a language model trained to autoregressively generate textual responses. This design enables strong progress on spoken dialogue and audio question answering, but it also introduces a fundamental supervision mismatch. The input contains rich frame-level acoustic structure, while the training objective provides signal only through sparse response tokens. The layers above the audio encoder are encouraged to retain only the information that is immediately useful for predicting the next text token, leaving many fine-grained acoustic cues weakly supervised or discarded.

This mismatch is especially pronounced for audio, which carries many properties that transcription alone cannot convey, including phonetic detail, speaker affect, background scene, prosody, and musical structure. One possible solution is to have the model verbalize intermediate thinking, as in discrete text chain-of-thought prompting [29, 52, 54], describing what the audio contains in natural language (for instance, "a man speaking in a high-pitched voice" or "the first note is a C") before producing the answer. However, many acoustic attributes cannot be serialized into natural language without losing fine-grained temporal and spectral detail. Faithful natural-language rationales for them

*Equal contribution.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).



(a) Vanilla (b) Discrete thinking (c) Continuous thinking (Ours)

Figure 1: **Thinking paradigms in audio language models.** (a) Vanilla audio LMs decode the response directly from audio and instruction tokens. (b) Discrete thinking generates textual thinking tokens autoregressively before the answer. (c) Continuous Audio Thinking (ours) prepends a fixed-length block of continuous thinking tokens that is consumed in a single prefill, letting the model think in an audio-aligned latent space without autoregressive cost.

are rarely available at scale, and even when they are, compressing low-level acoustic evidence into text introduces an unnecessary bottleneck.

These limitations motivate a new form of thinking that preserves and reorganizes acoustic information itself rather than verbalizing it. To meet this need, such thinking is required to unfold in continuous latent positions, remaining independent of text and free from supervision by natural-language rationales. It is further expected to serve as a workspace in which the language model maintains, aligns, and transforms acoustic information before committing to response generation.

Toward this, we propose **Continuous Audio Thinking** (**CoAT**), an auxiliary-supervision framework that equips audio language models with a latent workspace. CoAT places a thinking block between the user input and the assistant response, where the model organizes acoustic information before generating its reply. The thinking block is grounded by distillation from diverse audio experts covering reconstruction, speech content [41], sound events [30], paralinguistic features [35], and pitch [3], providing complementary acoustic dimensions that text supervision alone cannot supply. CoAT integrates with existing LALMs without architectural changes. Furthermore, CoAT requires neither textual rationales nor task-specific decoding formats, and can be learned from a modest amount of audio data. At inference, the thinking block is consumed in a single prefill, adding no autoregressive decoding cost over the baseline as shown in Figure 1.

We instantiate CoAT on three LALMs (Qwen2-Audio [9], Qwen2.5-Omni-7B [55], and Audio Flamingo 3 [19]) and evaluate it across diverse audio understanding and reasoning benchmarks. CoAT yields consistent gains across backbones and outperforms text chain-of-thought with substantially lower per-sample latency. Further analysis confirms that the auxiliary supervision propagates from the thinking positions to the model's textual responses. We summarize our contributions as follows:

- We propose a continuous-thinking paradigm for LALMs, enabling the model to organize acoustic information in the latent space without verbalization or autoregressive decoding.
- We propose a multi-expert distillation objective that grounds the thinking states in complementary acoustic dimensions, supplying signal that text supervision alone cannot provide.
- We evaluate CoAT on three LALMs and show that it consistently improves audio understanding and reasoning at lower latency than text chain-of-thought, with analysis showing that the supervision propagates from the thinking positions to the model's textual responses.

2 Related Work

Audio Language Models. A growing family of large language models has been adapted to natively process audio alongside text [9, 10, 13, 15, 16, 19, 20, 22, 31, 33, 47, 49, 55, 60]. A common architecture connects a pretrained audio encoder, typically an audio-text-aligned Whisper [44] encoder, to a language model via a lightweight projection layer, exposing audio as a stream of soft-token embeddings that are consumed by the decoder [13, 15, 16, 19, 20, 31, 49]. Another line trains a single multimodal model end-to-end so that audio, vision, and text share a unified token space [22, 55, 60]. In both cases, audio capabilities are typically obtained from supervised fine-tuning

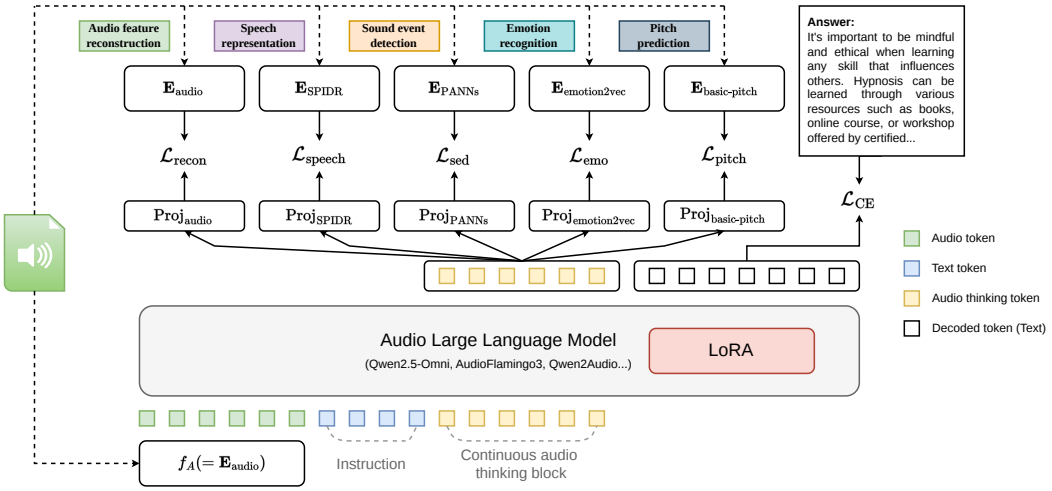


Figure 2: **CoAT architecture.** A continuous audio thinking block is supervised by five audio experts via per-task projection heads, covering audio feature reconstruction, speech representation, sound event detection, paralinguistic features, and pitch. The projection heads decode the shared hidden states into expert-aligned predictions, used only during training.

on triplets of audio, instruction, and target response, sometimes structured as a multi-stage curriculum that first aligns the audio encoder before instruction-tuning the language model [19, 55].

Continuous and Latent Thinking. Chain-of-thought prompting [29, 52, 54] demonstrated that explicit step-by-step traces in language tokens improve the reasoning behavior of large language models. Subsequent work has examined whether reasoning must remain in the discrete token space: Coconut [23] replaces textual chains with continuous embeddings that are looped back into the model, and Quiet-STaR [59] learns implicit thoughts that are scored against the next-token likelihood. A related thread treats internal recurrence and depth as an explicit reasoning resource via looped transformers and implicit chain-of-thought formulations [12, 17]. A concurrent line extends continuous reasoning beyond text: Chain-of-Visual-Thought [43] introduces continuous visual tokens that allow vision-language models to reason in a vision-aligned latent space, but it still autoregressively generates additional tokens for each task and pairs them with explicit textual reasoning, leaving the inference-cost limitations of token-by-token reasoning largely unaddressed.

Audio Encoders for Diverse Tasks. A wide range of audio encoders has been developed, each capturing a different aspect of the signal. Speech understanding is supported by self-supervised speech models that learn linguistic units from raw waveforms [2, 6, 24, 41]. General sound-event analysis builds on classifiers and masked models pretrained on broad acoustic taxonomies [7, 8, 26, 30]. Music understanding is addressed by encoders specialized for tonal and rhythmic structure [3, 57], while paralinguistic analysis relies on self-supervised models trained on emotional speech [35]. Neural audio codecs provide a complementary representation, compressing waveforms into compact latent sequences from which the original signal can be decoded [11, 58, 62]. We collect these complementary encoders into a single place and jointly utilize them in training time to enable an audio language model to think continuously, turning a scattered set of task-specific representations into a single audio-aware reasoning substrate.

3 Method

In this section, we propose Continuous Audio Thinking (CoAT), a method that allows LALMs to retain and organize acoustic content in latent form before generating their textual response. CoAT inserts a sequence of continuous latent positions between the audio input and the assistant response, and supervises the hidden states at those positions by distillation against multiple audio experts. The thinking block thus serves as a workspace grounded in complementary acoustic dimensions that text supervision alone cannot provide. We first formalize the audio language model interface (§3.1), then

introduce the thinking block (§3.2), the expert distillation (§3.3), and the stage-wise training schedule (§3.4). The overall pipeline of the proposed method is shown in Figure 2.

3.1 Large Audio Language Models

Before describing the method, we briefly outline how LALMs operate. We consider an audio language model composed of an LLM decoder f_L and an audio encoder f_A. The encoder f_A encodes a raw audio waveform a into a sequence of L_a audio token embeddings $\mathbf{x}_A \in \mathbb{R}^{L_a \times d}$ at a fixed rate r_s, where d is the hidden size of f_L. The input sequence is composed of a system prompt and a user prompt, $\mathbf{x} = [\mathbf{x}^{\text{sys}}, \mathbf{x}^{\text{usr}}]$, where the user prompt itself contains the audio token embeddings followed by a text prompt, $\mathbf{x}^{\text{usr}} = [\mathbf{x}_A, \mathbf{x}_{\text{txt}}]$. The model is trained with next-token prediction using the cross-entropy loss

$$\mathcal{L}_{\text{CE}}(\mathbf{x}) = -\sum_{t \in \mathcal{I}(\mathbf{x})} \log p_{f_L}(x_t \mid x_{<t}), \tag{1}$$

where $\mathcal{I}(\mathbf{x})$ indexes positions in the assistant response.

3.2 Continuous Audio Thinking Block

The thinking block introduces dedicated capacity for the model to process acoustic information before generating its response. We extend the model vocabulary with three special tokens, τ_s = <|audio_think_start|>, τ_p = <|audio_think|>, and τ_e = <|audio_think_end|>, that serve as the boundary and content positions of an audio thinking block. Given an input that contains an audio segment of length L_a, the corresponding thinking block is constructed by placing one τ_p for each audio token, framed by the boundary tokens,

$$\mathbf{b}(L_a) = \big[\, \tau_s, \underbrace{\tau_p, \ldots, \tau_p}_{L_a}, \tau_e \,\big]. \tag{2}$$

The block $\mathbf{b}(L_a)$ is appended to the input sequence, $\tilde{\mathbf{x}} = [\mathbf{x}^{\text{sys}}, \mathbf{x}^{\text{usr}}, \mathbf{b}(L_a)]$, and processed by decoder f_L. We collect the final-layer hidden states $\mathbf{H}_{\text{think}} \in \mathbb{R}^{L_a \times d}$ at the positions τ_p. At training time, the cross-entropy in Eq. (1) is computed only at response positions; the thinking-block tokens τ_s, τ_p, τ_e are excluded as next-token prediction targets. During inference, the same block is appended deterministically before generation, adding prefill cost without additional autoregressive decoding.

3.3 Distillation from Audio Experts

The thinking block provides space for processing acoustic information, but the text-only objective alone is insufficient for the model to learn to use this space effectively. We therefore distill frame-level features from audio experts to fill this gap. Let K denote the set of frozen audio experts used in CoAT. For each expert $k \in K$, an encoder $\mathbf{E}_k$ produces expert features $\mathbf{z}_k = \mathbf{E}_k(a) \in \mathbb{R}^{L_k \times e_k}$ from the raw audio a. For each expert $k \in K$, we attach a projection head P_k to the language model that projects $\mathbf{H}_{\text{think}}$ into expert-aligned predictions,

$$\hat{\mathbf{z}}_k = P_k(\mathbf{H}_{\text{think}}) \in \mathbb{R}^{L_a \times e_k}. \tag{3}$$

Each P_k is implemented as a single-block Transformer (multi-head attention and a feed-forward layer) followed by a linear map to the expert embedding dimension e_k. The experts fall into two families: representational experts and task-specialized experts, described in turn below.

3.3.1 Representational experts for location and content

Representational experts, comprising audio feature reconstruction and speech distillation, establish a foundation on which all subsequent supervision rests. Audio feature reconstruction anchors $\mathbf{H}_{\text{think}}$ to the audio encoder's latent space, so that the thinking block occupies the same subspace as the input acoustic representation. Speech distillation then enriches that representation with the linguistic structure captured by a self-supervised speech encoder, endowing the thinking block with both location and content before any application-specific objective is introduced.

Audio Feature Reconstruction. The expert encoder is the audio encoder of the backbone itself, $\mathbf{E}_{\text{audio}} = f_A$. The thinking block is trained to reproduce the same latent representation that f_L consumes for the audio, which constrains $\mathbf{H}_{\text{think}}$ to the audio-token subspace and supervises where acoustic information should be encoded before any semantic objective is introduced. We supervise this with a frame-wise MSE, $\mathcal{L}_{\text{recon}} = \text{MSE}(\hat{\mathbf{z}}_{\text{audio}}, \mathbf{z}_{\text{audio}})$.

Speech Representation Distillation. Furthermore, we employ $\mathbf{E}_{\text{SPIDR}}$ [41], a self-supervised speech encoder trained without labels to extract stable linguistic units from raw waveforms. We use its encoder output as the expert feature, supervised with a frame-wise MSE, $\mathcal{L}_{\text{speech}} = \text{MSE}(\hat{\mathbf{z}}_{\text{SPIDR}}, \mathbf{z}_{\text{SPIDR}})$. This task aligns the thinking representation with phonetic and lexical content, complementing the surface-level signal from audio feature reconstruction.

3.3.2 Task-specialized experts for application-domain capabilities

Building on this foundation, we attach experts that supply task-specialized capabilities. Each adds a capability that is otherwise difficult to acquire under text supervision alone, namely sound event detection for environmental audio, paralinguistic features for vocal affect, and pitch prediction for harmonic and prosodic structure. Importantly, this design is not specific to the three experts used here. Any audio encoder that captures a desired representation can be incorporated as an additional task.

Sound Event Detection. For sound-event semantics, we use the expert encoder $\mathbf{E}_{\text{PANNs}}$ [30], a CNN-based audio tagger pre-trained on AudioSet that produces frame-wise activations over 527 sound-event classes spanning speech, music, animal, vehicle, and ambient categories. We map the student prediction into the same class-logit space by passing it through PANNs's final classification head g_{cls}, and match it to the expert's per-frame, per-class probabilities with binary cross-entropy, $\mathcal{L}_{\text{sed}} = \text{BCE}(g_{\text{cls}}(\hat{\mathbf{z}}_{\text{PANNs}}), g_{\text{cls}}(\mathbf{z}_{\text{PANNs}}))$. This task exposes the thinking representation to a broad taxonomy of sound-event semantics, supporting general audio understanding tasks beyond speech.

Paralinguistic Feature Prediction. Voice affect is captured by $\mathbf{E}_{\text{emotion2vec}}$ [35], a self-supervised model trained on speech emotion data that produces frame-wise representations. We use its hidden-state output as the expert feature, supervised with a frame-wise MSE, $\mathcal{L}_{\text{emo}} = \text{MSE}(\hat{\mathbf{z}}_{\text{emotion2vec}}, \mathbf{z}_{\text{emotion2vec}})$. This task adds a non-lexical channel to the thinking representation that captures how the audio is spoken, such as affect, prosody, and intensity, rather than what is said.

Pitch Prediction. For pitch and harmonic structure, we adopt $\mathbf{E}_{\text{basic-pitch}}$ [3], a polyphonic pitch detector pre-trained on instrument transcription with a multi-pitch posteriorgram output. We use its intermediate convolutional activations as the expert feature; the dense intermediate representation preserves harmonic information while avoiding the sparsity of the final posteriorgram. The pitch task combines two losses, an MSE term on the intermediate feature and an auxiliary focal-BCE on the posteriorgram obtained by passing both student and expert features through the encoder's final convolution h. The loss is given by $\mathcal{L}_{\text{pitch}} = \text{MSE}(\hat{\mathbf{z}}_{\text{basic-pitch}}, \mathbf{z}_{\text{basic-pitch}}) + w^{\text{aux}}_{\text{pitch}} \text{focal-BCE}(h(\hat{\mathbf{z}}_{\text{basic-pitch}}), h(\mathbf{z}_{\text{basic-pitch}}))$. Pure BCE on the highly sparse posteriorgram collapses to all-zero predictions, while pure MSE on the dense intermediate feature under-constrains the pitch contour. This task introduces fine-grained pitch information into the thinking representation, which is otherwise sparsely covered by the speech- and sound-centric experts.

3.4 Training Objective

Stage-Wise Training. Training is organized as a sequence of S stages. Each stage is defined by its data, step budget, learning rate, and an active task subset $\mathcal{A}_s \subseteq K$ with per-task weights $w_k^{(s)}$, where s denotes the stage index. Each stage applies only the expert distillation tasks in $\mathcal{A}_s$ in addition to the language-modeling cross-entropy. The parameter scope spans LoRA [25] adapters, projection heads, added tokens, and the embedding layer. We instantiate $S = 2$ in our main experiments, with a warm-up stage $\mathcal{A}_1$ using audio feature reconstruction alone, followed by a multi-task stage with $\mathcal{A}_2$ spanning all experts.

Table 1: **Main results.** Per-benchmark performance of Qwen2-Audio, Qwen2.5-Omni-7B, and Audio Flamingo 3 with and without CoAT.

Benchmark	Eval	Qwen2-Audio	+ **CoAT**	Qwen2.5-Omni	+ **CoAT**	Audio Flamingo 3	+ **CoAT**
Audio Understanding & Reasoning							
General							
MMAU	Acc ↑	52.50	**66.90**	65.60	**69.40**	69.40	**70.00**
MMAR	Acc ↑	47.10	**52.60**	56.70	**62.20**	55.70	**59.60**
MMSU	Acc ↑	53.31	**58.27**	61.32	**64.99**	**60.01**	58.36
ClothoAQA	Acc ↑	75.52	**79.96**	84.67	**87.17**	80.10	**85.30**
Alpaca-Audio	GPT ↑	**57.40**	56.97	60.60	**64.24**	38.80	**58.59**
WavCaps	GPT ↑	32.20	**39.40**	29.40	**33.60**	**33.40**	29.00
AIR-Bench							
ABF Speech	Acc ↑	50.16	**70.98**	67.33	**75.99**	62.99	**71.24**
ABF Sound	Acc ↑	63.36	**71.64**	76.28	**76.49**	65.56	**69.50**
ABF Music	Acc ↑	51.23	**55.82**	58.20	**59.38**	58.62	**64.50**
Music Classification							
VocalSound	Acc ↑	88.10	**93.59**	91.36	**91.70**	**93.06**	92.39
GTZAN	Acc ↑	90.29	**92.39**	92.89	**92.99**	94.99	**95.50**
MuchoMusic	Acc ↑	69.25	**74.05**	72.87	**73.97**	81.63	**81.80**
Speech Emotion Recognition							
MELD	Acc ↑ / F1 ↑	25.5 / 21.8	**58.0 / 56.1**	49.4 / 51.1	**60.8 / 59.2**	40.8 / 45.9	**59.8 / 57.2**
IEMOCAP	Acc ↑	54.00	**72.70**	55.22	**79.01**	63.58	**70.39**
Automatic Speech Recognition							
LibriSpeech-clean	WER ↓	4.14	**2.30**	1.80	**1.77**	**1.57**	1.99
LibriSpeech-other		7.02	**4.74**	**3.40**	4.00	**3.13**	4.23
Common Voice 15		21.48	**9.25**	**7.60**	7.68	**7.40**	**7.40**
GigaSpeech		13.53	**10.52**	10.42	**9.42**	**10.27**	11.90
VoxPopuli		11.90	**7.03**	5.80	**5.56**	**5.55**	5.70
SPGISpeech		6.93	**2.86**	2.79	**2.10**	1.86	**1.84**
Switchboard		19.06	**9.25**	14.83	**7.35**	8.01	**7.18**

Overall Objective. The total loss at stage s is the sum of the language-modeling and distillation losses

$$\mathcal{L}^{(s)}_{\text{total}}(\tilde{\mathbf{x}}) = \mathcal{L}_{\text{CE}}(\tilde{\mathbf{x}}) + \sum_{k \in \mathcal{A}_s} w_k^{(s)} \mathcal{L}_k, \tag{4}$$

with $\mathcal{L}_{\text{CE}}$ computed on $\tilde{\mathbf{x}}$ under the thinking-token mask. The thinking block thus contributes purely through the distillation losses, leaving the model's text generation unchanged.

4 Experiments

4.1 Experimental Setup

We instantiate CoAT on three pretrained audio language models: Qwen2-Audio [9], Qwen2.5-Omni-7B [55], and Audio Flamingo 3 [19]. All backbones are trained with the same two-stage CoAT schedule: a reconstruction-only warm-up stage followed by a multi-task stage with all five experts active.

We use the same public training mixture, sampling policy, and evaluation protocol across all backbones and benchmarks. The training mixture covers automatic speech recognition [1, 5, 18, 39, 40, 50], audio and speech question answering [4, 28, 32, 63], audio captioning [14, 28], multiple-choice question audio understanding [4, 42, 63], music understanding [38], spoken-instruction following [46], and a small text-only supervised fine-tuning split [27]. Further experimental details are provided in Appendix A.

Table 2: **Comparison with discrete reasoning.** Reasoning accuracy and per-sample inference cost on the full MMAU and MMAR evaluation sets.

	MMAU ↑	MMAR ↑	TTFT (s) ↓	Dec. time (s) ↓	Dec. tok	Total (s) ↓
Qwen2.5-Omni	65.60	56.70	0.124	0.014	6.5	0.139
w/ text-CoT (prompt)	66.90	56.70	0.133 (1.07×)	0.147 (10.5×)	59.5 (9.15×)	0.280 (2.01×)
w/ CoAT (ours)	69.40	62.20	0.126 (1.01×)	0.019 (1.34×)	8.5 (1.31×)	0.145 (1.04×)
Audio Flamingo 3	69.40	55.70	0.034	0.004	2.2	0.038
w/ text-CoT	64.52	54.25	0.034 (1.00×)	0.205 (55.3×)	80.4 (36.5×)	0.239 (6.24×)
w/ CoAT (ours)	70.00	59.60	0.037 (1.07×)	0.028 (7.57×)	11.7 (5.32×)	0.065 (1.70×)

4.2 Main Results

Across all three backbones, CoAT improves the pretrained baseline on the majority of benchmarks, as reported in Table 1, demonstrating its generality across various models and tasks. The improvements are most pronounced on understanding- and reasoning-intensive tasks such as MELD, MMAR, and Alpaca-Audio, indicating that CoAT is particularly effective on audio understanding and reasoning On automatic speech recognition, CoAT substantially improves the weaker Qwen2-Audio backbone while preserving performance on Qwen2.5-Omni and Audio Flamingo 3. Overall, CoAT generalizes across heterogeneous audio-language backbones and yields the largest gains on reasoning-heavy audio understanding.

4.3 Comparison with Discrete Thinking

We compare CoAT's continuous thinking approach against the discrete-thinking alternative where the model autoregressively generates a textual reasoning trace before producing the answer. For Audio Flamingo 3, which has a native think model, we use built-in chain-of-thought template. For Qwen2.5-Omni, which has no native think mode, we instead induce text-CoT through a prompt-level instruction asking the model to reason step by step.

Table 2 reports reasoning accuracy on two benchmarks (MMAU and MMAR) with four inference-cost metrics: time to first token (TTFT), decoding time (Dec. time), the number of decoded tokens (Dec. tok), and end-to-end latency (Total). Each reported value is the mean over multiple runs measured in the same environment; full experimental details are provided in Appendix C.

Text-CoT and CoAT differ in where they spend inference cost. Text-CoT autoregressively decodes a reasoning chain before the answer, so its overhead lands almost entirely in the decode stage and does not amortize across batched requests. CoAT instead consumes its thinking block in a single prefill, shifting the overhead out of the decoding phase. As a result, CoAT runs much faster than text-CoT while improving reasoning accuracy over the same-backbone baseline. Appendix D provides a full per-benchmark comparison with Audio Flamingo 3's native think mode.

4.4 Analysis

We probe whether CoAT's auxiliary supervision injects task-relevant information at thinking positions by training linear probes on the LM hidden state at two positions. The first one is the audio-think hidden, taken as the mean over the thinking block τ_p where CoAT's supervision attaches. The second one is the pre-generation hidden, taken from the last τ_e token for CoAT and from the last input token before the assistant turn for the vanilla baseline. Both correspond to the model's decision-time representation.

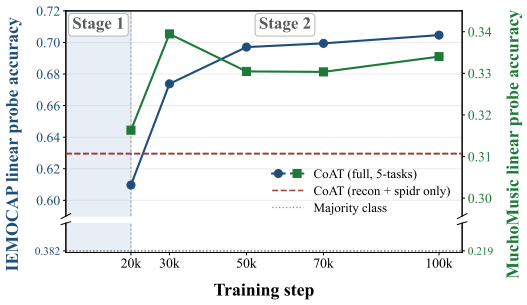
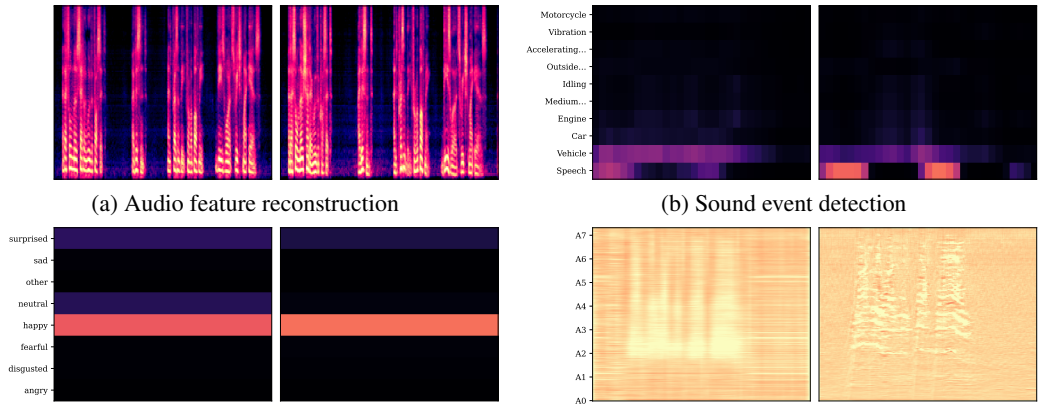


Figure 3: Linear probe accuracy at the audio-think hidden across training checkpoints, on 4-class IEMOCAP emotion and 12-class MuchoMusic dominant pitch.

Linear probing accuracy across training checkpoints. We probe two targets aligned with the training supervision, specifically 4-class



(a) Audio feature reconstruction

(b) Sound event detection

(c) Paralinguistic feature prediction

(d) Pitch prediction

Figure 4: **Example reconstructions from CoAT's per-task heads.** Each pair shows the expert target (right) and the corresponding student prediction (left) at the audio-think positions.

emotion on IEMOCAP and 12-class dominant pitch on MuchoMusic. Linear probes are trained on the audio-think hidden of the full 5-head CoAT across training checkpoints. Figure 3 shows that probe accuracy rises in stage 2 on both targets, when specialized experts begin contributing supervision beyond reconstruction. This indicates that the auxiliary supervision injects task-relevant information into the supervised position.

Correlation between thinking-token information and downstream performance. We now probe two targets aligned with the downstream task, namely 4-class emotion on IEMOCAP and 7-class instrument family on the MuchoMusic instrument-question subset. We compare three models: the vanilla Qwen2.5-Omni baseline, a CoAT control learned only by representational experts, and the CoAT trained with all five experts. Table 3 reports probe accuracy alongside within-model Spearman correlations between probe confidence and downstream task performance. The CoAT trained with all five experts attains the highest probe accuracy and the strongest within-model correlation across all cases. Together, the supervision injects information at the thinking position that accumulates over training and correlates with downstream task performance.

Table 3: Probe accuracy and Spearman ρ between probe confidence and downstream task performance, reported as (accuracy/ρ). *Think* and *Pre-gen* denote the audio-think and pre-generation hidden states, respectively.

	IEMOCAP		MuchoMusic	
Model	Think	Pre-gen	Think	Pre-gen
Qwen2.5-Omni	–	.622/.32	–	.510/.06
CoAT (recon+spidr)	.630/.36	.707/.57	.301/.20	.567/.18
CoAT (full)	**.705/.49**	**.727/.56**	**.324/.21**	**.607/.22**

Visualizing per-head reconstructions. To verify that the thinking representation distills information from each expert, we visualize example reconstructions in Figure 4. For audio feature reconstruction, we use the Sim-Whisper [62] codec decoder to reconstruct audio from the predicted feature, which we visualize as a mel-spectrogram. For the other tasks, we pass the predicted feature through the expert's prediction head and visualize the resulting output. Each task's reconstruction faithfully matches the expert target, confirming that the thinking representation encodes the information required to perform every supervised task.

4.5 Ablation Studies

We conduct ablations on a single backbone, Qwen2.5-Omni, to validate our design choices.

Task Ablation. Table 4 reports the cumulative contribution of each component. Supervised fine-tuning on top of Qwen2.5-Omni substantially improves Emotion and reduces ASR error, but leaves General and AIR-Bench essentially flat and regresses Music below the baseline. Adding the continuous thinking block improves General, Emotion, and ASR, and partially recovers Music, though

Table 4: **Task ablation.** Starting from the baseline model, we cumulatively add one component at a time and retrain under the same schedule: supervised fine-tuning (SFT), continuous thinking block, representational expert distillation, and specialized expert distillation. We report the average performance of benchmarks with the same metric.

Variant	General (Acc ↑)	AIR-Bench (Acc ↑)	Music (Acc ↑)	Emotion (Acc ↑)	ASR (WER ↓)
Qwen2.5-Omni	67.07	67.27	85.71	52.29	6.66
+ SFT	67.48	67.33	84.65	67.31	5.58
+ Continuous Thinking Block	68.52	67.01	84.86	67.14	5.42
+ Representational Expert Distillation	69.89	68.75	84.39	67.32	5.54
+ Specialized Expert Distillation	**70.94**	**70.62**	**86.22**	**69.91**	**5.41**

Table 5: **Projector type.** We compare a linear projection against the single-block Transformer projector used in the main results. We report the average performance of benchmarks with the same metric.

Projector	General (Acc ↑)	AIR-Bench (Acc ↑)	Music (Acc ↑)	Emotion (Acc ↑)	ASR (WER ↓)
Linear	70.08	**70.75**	83.86	69.37	5.50
Transformer (ours)	**70.94**	70.62	**86.22**	**69.91**	**5.41**

AIR-Bench and Music remain below the original Qwen2.5-Omni baseline. Representational expert distillation lifts AIR-Bench substantially, further recovers Music, and raises General. Specialized expert distillation attains the best score on every metric, including the only Music value that surpasses the original baseline. Relative to SFT alone, CoAT improves every metric and is the only configuration that lifts both broad reasoning and music understanding above the original Qwen2.5-Omni baseline. These results show that the improvements are not driven by SFT alone: representational expert distillation accounts for the gains in general understanding and reasoning, while specialized expert distillation yields the largest improvements on task-relevant metrics.

Projector Type. The choice of projector affects how the thinking representation is decoded into expert-aligned predictions. We compare a linear projector against the single-block Transformer projector used in our main results. Table 5 shows that the Transformer projector generally outperforms the simple linear projection layer.

5 Conclusion

We introduce Continuous Audio Thinking (CoAT), a framework that equips audio language models with a continuous latent workspace for organizing acoustic information before response generation. Audio experts supervise this workspace through distillation under a representational-then-specialized schedule: latent states are first anchored to the audio space and then aligned with semantic, paralinguistic, and musical structure. CoAT consistently improves audio understanding and reasoning across three LALMs, while running at lower per-sample latency than text chain-of-thought. Further analyses show that the auxiliary signal propagates from the thinking positions to the textual outputs: linear probes on those positions became more accurate over training, and within-model probe confidence predicted downstream task performance. CoAT shows that continuous latent thinking can support reasoning in modalities that are difficult to verbalize.

Limitations and Future Work

Our study has two main limitations, which we leave to future work. First, the thinking block in CoAT is deterministic, occupying a fixed span at a pre-defined position between the audio input and the assistant response. The model does not learn when or how long to think, nor does it interleave thinking with response generation. CoAT thus realizes a latent workspace but not multi-step latent reasoning, and extending it to dynamic or interleaved thinking blocks is an important next step. Second, our empirical validation is confined to the audio domain. Although the proposed mechanism is modality-agnostic in principle, whether the same recipe transfers to vision- and video-language models remains an open question.

Acknowledgements

This work was partly supported by the InnoCORE program of the Ministry of Science and ICT (AI Meta-Scientist, N10260110) and Artificial intelligence industrial convergence cluster development project funded by the Ministry of Science and ICT(MSIT, Korea)&Gwangju Metropolitan City. All experiments were conducted on Cosmo, PYLER's internal GPU cluster.

References

[1] Rosana Ardila, Megan Branson, Kelly Davis, Michael Henretty, Michael Kohler, Josh Meyer, Reuben Morais, Lindsay Saunders, Francis M. Tyers, and Gregor Weber. Common voice: A massively-multilingual speech corpus, 2020.

[2] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in neural information processing systems*, 33:12449–12460, 2020.

[3] Rachel M. Bittner, Juan José Bosch, David Rubinstein, Gabriel Meseguer-Brocal, and Sebastian Ewert. A lightweight instrument-agnostic model for polyphonic note transcription and multipitch estimation. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Singapore, 2022.

[4] Carlos Busso, Murtaza Bulut, Chi-Chun Lee, Abe Kazemzadeh, Emily Mower, Samuel Kim, Jeannette N Chang, Sungbok Lee, and Shrikanth S Narayanan. Iemocap: Interactive emotional dyadic motion capture database. *Language resources and evaluation*, 42(4):335–359, 2008.

[5] Guoguo Chen, Shuzhou Chai, Guanbo Wang, Jiayu Du, Wei-Qiang Zhang, Chao Weng, Dan Su, Daniel Povey, Jan Trmal, Junbo Zhang, et al. Gigaspeech: An evolving, multi-domain asr corpus with 10,000 hours of transcribed audio. *arXiv preprint arXiv:2106.06909*, 2021.

[6] Sanyuan Chen, Chengyi Wang, Zhengyang Chen, Yu Wu, Shujie Liu, Zhuo Chen, Jinyu Li, Naoyuki Kanda, Takuya Yoshioka, Xiong Xiao, et al. Wavlm: Large-scale self-supervised pre-training for full stack speech processing. *IEEE Journal of Selected Topics in Signal Processing*, 16(6):1505–1518, 2022.

[7] Sanyuan Chen, Yu Wu, Chengyi Wang, Shujie Liu, Daniel Tompkins, Zhuo Chen, Wanxiang Che, Xiangzhan Yu, and Furu Wei. Beats: Audio pre-training with acoustic tokenizers. In *International Conference on Machine Learning*, pages 5178–5193. PMLR, 2023.

[8] Wenxi Chen, Yuzhe Liang, Ziyang Ma, Zhisheng Zheng, and Xie Chen. Eat: self-supervised pre-training with efficient audio transformer. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pages 3807–3815, 2024.

[9] Yunfei Chu, Jin Xu, Qian Yang, Haojie Wei, Xipin Wei, Zhifang Guo, Yichong Leng, Yuanjun Lv, Jinzheng He, Junyang Lin, et al. Qwen2-audio technical report. *arXiv preprint arXiv:2407.10759*, 2024.

[10] Yunfei Chu, Jin Xu, Xiaohuan Zhou, Qian Yang, Shiliang Zhang, Zhijie Yan, Chang Zhou, and Jingren Zhou. Qwen-audio: Advancing universal audio understanding via unified large-scale audio-language models. *arXiv preprint arXiv:2311.07919*, 2023.

[11] Alexandre Défossez, Jade Copet, Gabriel Synnaeve, and Yossi Adi. High fidelity neural audio compression. *arXiv preprint arXiv:2210.13438*, 2022.

[12] Yuntian Deng, Yejin Choi, and Stuart Shieber. From explicit cot to implicit cot: Learning to internalize cot step by step. *arXiv preprint arXiv:2405.14838*, 2024.

[13] Soham Deshmukh, Benjamin Elizalde, Rita Singh, and Huaming Wang. Pengi: An audio language model for audio tasks. *Advances in Neural Information Processing Systems*, 36:18090–18108, 2023.

[14] Konstantinos Drossos, Samuel Lipping, and Tuomas Virtanen. Clotho: An audio captioning dataset. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 736–740. IEEE, 2020.

[15] 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. *arXiv preprint arXiv:2503.03983*, 2025.

[16] Sreyan Ghosh, Sonal Kumar, Ashish Seth, Chandra Kiran Reddy Evuru, Utkarsh Tyagi, S Sakshi, Oriol Nieto, Ramani Duraiswami, and Dinesh Manocha. Gama: A large audio-language model with advanced audio understanding and complex reasoning abilities. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 6288–6313, 2024.

[17] Angeliki Giannou, Shashank Rajput, Jy-yong Sohn, Kangwook Lee, Jason D Lee, and Dimitris Papailiopoulos. Looped transformers as programmable computers. In *International Conference on Machine Learning*, pages 11398–11442. PMLR, 2023.

[18] John J Godfrey, Edward C Holliman, and Jane McDaniel. Switchboard: Telephone speech corpus for research and development. In *[Proceedings] ICASSP-92: 1992 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 517–520. IEEE, 1992.

[19] Arushi Goel, Sreyan Ghosh, Jaehyeon Kim, Sonal Kumar, Zhifeng Kong, Sang-gil Lee, Chao-Han Huck Yang, Ramani Duraiswami, Dinesh Manocha, Rafael Valle, et al. Audio flamingo 3: Advancing audio intelligence with fully open large audio language models. *arXiv preprint arXiv:2507.08128*, 2025.

[20] Yuan Gong, Hongyin Luo, Alexander H Liu, Leonid Karlinsky, and James Glass. Listen, think, and understand. *arXiv preprint arXiv:2305.10790*, 2023.

[21] Yuan Gong, Jin Yu, and James Glass. Vocalsound: A dataset for improving human vocal sounds recognition. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 151–155. IEEE, 2022.

[22] Jiaming Han, Kaixiong Gong, Yiyuan Zhang, Jiaqi Wang, Kaipeng Zhang, Dahua Lin, Yu Qiao, Peng Gao, and Xiangyu Yue. Onellm: One framework to align all modalities with language. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26584–26595, 2024.

[23] Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space. *arXiv preprint arXiv:2412.06769*, 2024.

[24] Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM transactions on audio, speech, and language processing*, 29:3451–3460, 2021.

[25] Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

[26] Po-Yao Huang, Hu Xu, Juncheng Li, Alexei Baevski, Michael Auli, Wojciech Galuba, Florian Metze, and Christoph Feichtenhofer. Masked autoencoders that listen. *Advances in neural information processing systems*, 35:28708–28720, 2022.

[27] Liwei Jiang, Kavel Rao, Seungju Han, Allyson Ettinger, Faeze Brahman, Sachin Kumar, Niloofar Mireshghallah, Ximing Lu, Maarten Sap, Yejin Choi, and Nouha Dziri. Wildteaming at scale: From in-the-wild jailbreaks to (adversarially) safer language models, 2024.

[28] Chris Dongjoo Kim, Byeongchang Kim, Hyunmin Lee, and Gunhee Kim. Audiocaps: Generating captions for audios in the wild. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 119–132, 2019.

[29] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213, 2022.

[30] Qiuqiang Kong, Yin Cao, Turab Iqbal, Yuxuan Wang, Wenwu Wang, and Mark D Plumbley. Panns: Large-scale pretrained audio neural networks for audio pattern recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:2880–2894, 2020.

[31] 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. *arXiv preprint arXiv:2402.01831*, 2024.

[32] Samuel Lipping, Parthasaarathy Sudarsanam, Konstantinos Drossos, and Tuomas Virtanen. Clotho-aqa: A crowdsourced dataset for audio question answering. In *2022 30th European Signal Processing Conference (EUSIPCO)*, pages 1140–1144. IEEE, 2022.

[33] Shansong Liu, Atin Sakkeer Hussain, Chenshuo Sun, and Ying Shan. Music understanding llama: Advancing text-to-music generation with question answering and captioning. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 286–290. IEEE, 2024.

[34] Ziyang Ma, Yinghao Ma, Yanqiao Zhu, Chen Yang, Yi-Wen Chao, Ruiyang Xu, et al. Mmar: A challenging benchmark for deep reasoning in speech, audio, music, and their mix. *arXiv preprint arXiv:2505.13032*, 2025.

[35] Ziyang Ma, Zhisheng Zheng, Jiaxin Ye, Jinchao Li, Zhifu Gao, Shiliang Zhang, and Xie Chen. emotion2vec: Self-supervised pre-training for speech emotion representation. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 15747–15760, 2024.

[36] Navonil Majumder, Chia-Yu Hung, Deepanway Ghosal, Wei-Ning Hsu, Rada Mihalcea, and Soujanya Poria. Tango 2: Aligning diffusion-based text-to-audio generations through direct preference optimization, 2024.

[37] Xinhao Mei, Chutong Meng, Haohe Liu, Qiuqiang Kong, Tom Ko, Chengqi Zhao, Mark D Plumbley, Yuexian Zou, and Wenwu Wang. Wavcaps: A chatgpt-assisted weakly-labelled audio captioning dataset for audio-language multimodal research. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 32:3339–3354, 2024.

[38] Jan Melechovsky, Zixun Guo, Deepanway Ghosal, Navonil Majumder, Dorien Herremans, and Soujanya Poria. Mustango: Toward controllable text-to-music generation. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 8286–8309, 2024.

[39] Patrick K. O'Neill, Vitaly Lavrukhin, Somshubra Majumdar, Vahid Noroozi, Yuekai Zhang, Oleksii Kuchaiev, Jagadeesh Balam, Yuliya Dovzhenko, Keenan Freyberg, Michael D. Shulman, Boris Ginsburg, Shinji Watanabe, and Georg Kucsko. Spgispeech: 5,000 hours of transcribed financial audio for fully formatted end-to-end speech recognition, 2021.

[40] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: an asr corpus based on public domain audio books. In *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5206–5210. IEEE, 2015.

[41] Maxime Poli, Mahi Luthra, Youssef Benchekroun, Yosuke Higuchi, Martin Gleize, Jiayi Shen, Robin Algayres, Yu-An Chung, Mido Assran, Juan Pino, and Emmanuel Dupoux. Spidr: Learning fast and stable linguistic units for spoken language models without supervision. *Transactions on Machine Learning Research*, 2025.

[42] Soujanya Poria, Devamanyu Hazarika, Navonil Majumder, Gautam Naik, Erik Cambria, and Rada Mihalcea. Meld: A multimodal multi-party dataset for emotion recognition in conversations. In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pages 527–536, 2019.

[43] Yiming Qin, Bomin Wei, Jiaxin Ge, Konstantinos Kallidromitis, Stephanie Fu, Trevor Darrell, and XuDong Wang. Chain-of-visual-thought: Teaching vlms to see and think better with continuous visual tokens. *arXiv preprint arXiv:2511.19418*, 2025.

[44] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine Mcleavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 28492–28518. PMLR, 23–29 Jul 2023.

[45] 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.

[46] Min-Han Shih, Ho-Lam Chung, Yu-Chi Pai, Ming-Hao Hsu, Guan-Ting Lin, Shang-Wen Li, and Hung-yi Lee. Gsqa: An end-to-end model for generative spoken question answering. *arXiv preprint arXiv:2312.09781*, 2023.

[47] Yu Shu, Siwei Dong, Guangyao Chen, Wenhao Huang, Ruihua Zhang, Daochen Shi, Qiqi Xiang, and Yemin Shi. Llasm: Large language and speech model. *arXiv preprint arXiv:2308.15930*, 2023.

[48] Bob L Sturm. The gtzan dataset: Its contents, its faults, their effects on evaluation, and its future use. *arXiv preprint arXiv:1306.1461*, 2013.

[49] Changli Tang, Wenyi Yu, Guangzhi Sun, Xianzhao Chen, Tian Tan, Wei Li, Lu Lu, Zejun Ma, and Chao Zhang. Salmonn: Towards generic hearing abilities for large language models. *arXiv preprint arXiv:2310.13289*, 2023.

[50] Changhan Wang, Morgane Rivière, Ann Lee, Anne Wu, Chaitanya Talnikar, Daniel Haziza, Mary Williamson, Juan Pino, and Emmanuel Dupoux. Voxpopuli: A large-scale multilingual speech corpus for representation learning, semi-supervised learning and interpretation, 2021.

[51] Dingdong Wang, Jincenzi Wu, Junan Li, Dongchao Yang, Xueyuan Chen, Tianhua Zhang, and Helen Meng. Mmsu: A massive multi-task spoken language understanding and reasoning benchmark. *arXiv preprint arXiv:2506.04779*, 2025.

[52] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

[53] Benno Weck, Ilaria Manco, Emmanouil Benetos, Elio Quinton, György Fazekas, and Dmitry Bogdanov. Muchomusic: Evaluating music understanding in multimodal audio-language models. In *Proceedings of the 25th International Society for Music Information Retrieval Conference (ISMIR)*, 2024.

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

[55] 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. Qwen2.5-omni technical report. *arXiv preprint arXiv:2503.20215*, 2025.

[56] Qian Yang, Jin Xu, Wenrui Liu, Yunfei Chu, Ziyue Jiang, Xiaohuan Zhou, Yichong Leng, Yuanjun Lv, Zhou Zhao, Chang Zhou, et al. Air-bench: Benchmarking large audio-language models via generative comprehension. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1979–1998, 2024.

[57] LI Yizhi, Ruibin Yuan, Ge Zhang, Yinghao Ma, Xingran Chen, Hanzhi Yin, Chenghao Xiao, Chenghua Lin, Anton Ragni, Emmanouil Benetos, et al. Mert: Acoustic music understanding model with large-scale self-supervised training. In *The Twelfth International Conference on Learning Representations*, 2024.

[58] Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, and Marco Tagliasacchi. Soundstream: An end-to-end neural audio codec. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:495–507, 2021.

[59] Eric Zelikman, Georges Harik, Yijia Shao, Varuna Jayasiri, Nick Haber, and Noah D Goodman. Quiet-star: Language models can teach themselves to think before speaking. *arXiv preprint arXiv:2403.09629*, 2024.

[60] Jun Zhan, Junqi Dai, Jiasheng Ye, Yunhua Zhou, Dong Zhang, Zhigeng Liu, Xin Zhang, Ruibin Yuan, Ge Zhang, Linyang Li, et al. Anygpt: Unified multimodal llm with discrete sequence modeling. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9637–9662, 2024.

[61] Kaichen Zhang, Bo Li, Peiyuan Zhang, Fanyi Pu, Joshua Adrian Cahyono, Kairui Hu, Shuai Liu, Yuanhan Zhang, Jingkang Yang, Chunyuan Li, and Ziwei Liu. Lmms-eval: Reality check on the evaluation of large multimodal models. *arXiv preprint arXiv:2407.12772*, 2024.

[62] Xin Zhang, Lin Li, Xiangni Lu, Jianquan Liu, and Kong Aik Lee. Speaking clearly: A simplified whisper-based codec for low-bitrate speech coding. In *ICASSP 2026-2026 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 17037–17041. IEEE, 2026.

[63] Zihan Zhao, Yiyang Jiang, Heyang Liu, Yu Wang, and Yanfeng Wang. Librisqa: A novel dataset and framework for spoken question answering with large language models. *IEEE Transactions on Artificial Intelligence*, 2024.

Table A: Audio expert encoders used in CoAT. e_k is the expert embedding dimension and r_k is the frame rate at which the expert emits features.

Expert	Aspect	e_k	r_k	Loss
Sim-Whisper [62]	self-reconstruction	1280	25 Hz	MSE
SPIDR [41]	speech	768	50 Hz	MSE
PANNs [30]	sound events	2048	25 Hz	BCE
emotion2vec [35]	paralinguistic	1024	50 Hz	MSE
basic_pitch [3]	pitch	2112	86 Hz	MSE + Focal BCE

Table B: **Training schedule.** CoAT is trained in two stages that share the same optimizer, learning rate, and batch configuration. Only the active expert set and the per-expert loss weights change. Stage 1 is a reconstruction-only warm-up, and stage 2 activates all five experts. The basic_pitch expert carries two coupled losses, an MSE on its dense intermediate feature and an auxiliary focal-BCE on the 264-bin pitch posteriorgram.

	Stage 1 (warm-up)	Stage 2 (multi-task)
Schedule and optimization		
Steps	20,000	80,000
Learning rate	5×10^{-5}	5×10^{-5}
LR schedule	linear, warmup 0.03	linear, warmup 0.03
Optimizer	AdamW	AdamW
Precision	bf16	bf16
Per-device batch / grad-accum	4/1	4/1
Effective batch (4 GPUs)	16	16
Max sequence length	2048	2048
LoRA r / α / dropout	16/32/0.05	16/32/0.05
Expert dropout p_{drop}	–	0.5
Active experts and loss weights w_k		
w_{recon} (Sim-Whisper, MSE)	5.0	2.0
w_{spidr} (SPIDR, MSE)	–	0.2
w_{uss} (PANNs USS, BCE)	–	100
w_{emotion} (emotion2vec, MSE)	–	40
w_{pitch} (basic_pitch feature, MSE)	–	5.0
$w^{\text{aux}}_{\text{pitch}}$ (basic_pitch posteriorgram, focal-BCE)	–	8

A Experimental Details

Training configuration. All three backbones produce audio token embeddings at $r_s = 25$ Hz, with LM hidden size $d = 4096$ for Qwen2-Audio and $d = 3584$ for Qwen2.5-Omni-7B and Audio Flamingo 3. Throughout the main CoAT training runs, the audio tower and all expert encoders are kept frozen. Gradients update only the LoRA adapters and the per-task projection heads. We use LoRA adapters of rank 16 with $\alpha = 32$. Each projection head P_k is a single-block Transformer with multi-head attention, a feed-forward layer, and a linear map to the corresponding expert embedding dimension.

CoAT is trained with a two-stage schedule. The first stage is a reconstruction-only warm-up that aligns the thinking states with the audio-token distribution. The second stage activates all five expert losses together with the language-modeling loss. The loss weight for each expert compensates for differences in output scale, and the same optimizer, learning rate, LoRA configuration, and training schedule are used for every backbone. Training runs on 4× NVIDIA B200 GPUs with an effective batch size of 16, taking approximately 88 B200 GPU-hours per backbone for the full two-stage schedule. The full evaluation suite requires approximately 15 B200 GPU-hours per evaluated model. Table A lists the expert encoders, and Table B lists the training schedule and hyperparameters.

Training datasets. Training data are drawn from publicly available sources organized into seven task groups: automatic speech recognition, audio and speech question answering, audio captioning, multiple-choice audio understanding, music understanding, spoken-instruction following, and a small

Table C: Training corpus and single-stage sampling budget (1.6M target samples). Pool is the unique row count per source. Subtotal aggregates the pool within each category. Share is the per-category mixing ratio; Samples is the resulting sample budget.

Category	Source dataset	Pool	Subtotal	Share	Samples
Automatic Speech Recognition	LibriSpeech	281,238	2,699,934	21.25%	340,000
	GigaSpeech	910,122			
	CommonVoice 15 (en)	1,070,055			
	VoxPopuli (en)	182,466			
	Switchboard	178,980			
	SPGISpeech	77,073			
Audio & speech QA	OpenASQA–AudioCaps	534,473	758,560	21.25%	340,000
	OpenASQA–IEMOCAP	108,497			
	LibriSQA (open)	103,928			
	Clotho-AQA	11,662			
Audio captioning	AudioCaps	45,178	51,107	14.00%	224,000
	Clotho-v2	5,929			
Multiple-choice audio understanding	MELD (emotion MCQ)	9,988	121,677	23.50%	376,000
	IEMOCAP (emotion MCQ)	7,676			
	LibriSQA (MCQ)	104,013			
Music understanding	MusicBench (caption)	52,768	105,536	10.00%	160,000
	MusicBench (MCQ)	52,768			
Spoken-instruction following	GSQA	51,349	51,349	5.00%	80,000
Text-only SFT	WildJailbreak	261,538	261,538	5.00%	80,000
Total			**4,049,701**	**100.00%**	**1,600,000**

text-only supervised fine-tuning split. The text-only split is included to preserve instruction-following ability and to prevent catastrophic over-refusal on harmful prompts, we observed that both abilities degraded rapidly under audio-only multi-task training. After fixing the per-task sampling ratio, we draw a training subset sized so that the full step schedule corresponds to one epoch over the subset. To prevent overfitting on smaller tasks, we additionally cap the maximum number of epochs that any single task can repeat within this subset.

Table C lists the full training corpus together with the sampling ratios used to materialize a single stage of 1,600,000 shuffled rows. Each category draws independently from its constituent sources at uniform probability, so per source budgets within a category scale with pool size. The number of optimization steps is configured so that effective batch size multiplied by total steps matches the stage size, giving each row in the materialized file approximately one expected pass per stage and avoiding within stage oversampling. The spoken instruction following category pools the evaluation aligned GSQA prompts and the text only WildJailbreak split is included to anchor refusal behavior on harmful prompts while preventing over refusal on benign adversarial ones.

Evaluation suite. We evaluate CoAT on a broad benchmark suite organized into five families that mirror the rows of Table 1. General audio understanding and reasoning uses MMAU [45], MMAR [34], MMSU [51], ClothoAQA [32], Alpaca-Audio [36], and WavCaps [37]. AIR-Bench Foundation [56] contributes its three subsets covering speech, sound, and music. Music classification spans VocalSound [21], GTZAN [48], and MuchoMusic [53]. Speech emotion recognition is measured on MELD [42] and IEMOCAP [4]. Speech transcription uses LibriSpeech [40], Common Voice 15 [1], GigaSpeech [5], VoxPopuli [50], SPGISpeech [39], and Switchboard [18]. MELD reports both accuracy and class-support-weighted F1, following the convention of the MELD paper for class-imbalanced 7-way emotion classification.

Table D: **Per-benchmark evaluation protocol.** M denotes `max_new_tokens` as configured in `lmms-eval`. LLM judges use the official prompt of each benchmark, reproduced verbatim.

Family	Benchmark	Split	M	Metric / Judge
General Audio	MMAU	test_mini	128	Accuracy
	MMAR	test	128	Accuracy
	MMSU	test	256	Accuracy
	ClothoAQA	test	8	Exact match
	Alpaca-Audio	test	1024	GPT-4o
	WavCaps	test	1024	GPT-4o
AIR-Bench Foundation	ABF Speech	foundation	256	Accuracy
	ABF Sound	foundation	256	Accuracy
	ABF Music	foundation	256	Accuracy
Music classification	VocalSound	test	20	Accuracy
	GTZAN	test	20	Accuracy
	MuchoMusic	test	256	Accuracy
Speech emotion recognition	MELD	test	20	Accuracy / F1
	IEMOCAP	Session 5 (LOSO)	20	Accuracy
Automatic Speech Recognition	LibriSpeech-clean / -other	test	256	WER ↓
	Common Voice 15	test (en)	256	WER ↓
	GigaSpeech	test	256	WER ↓
	VoxPopuli	test (en)	2048	WER ↓
	SPGISpeech	test	512	WER ↓
	Switchboard	eval2000	512	WER ↓

B Evaluation Protocol Details

All inference uses vLLM as the backend with greedy decoding ($T = 0$, top-$p = 0.95$). For tasks supplied by upstream `lmms-eval` [61] we adopt the released implementation as-is. For benchmarks that are either not natively supported by upstream or whose upstream implementation we found to mis-handle the audio, the prompt template, or the metric, we use our own task definitions. These custom definitions cover IEMOCAP, MELD, SPGISpeech, Switchboard, GTZAN, and VocalSound.

MELD's seven emotion classes are highly imbalanced, with neutral alone accounting for roughly half of the test set, so accuracy alone can be inflated by a model that always predicts the majority class. We therefore report a weighted F1 score, defined as the average of per-class F1 with each class weighted by its support (*i.e.*, the number of test samples in that class).

Audio clips longer than 120 seconds are skipped at data-loading. This affects only a negligible fraction of inputs and is applied identically to every model, so it does not affect relative comparisons. GigaSpeech references additionally contain meta-tags (`<MUSIC>`, `<NOISE>`, `<SIL>`, `<OTHER>`) that are stripped before scoring. Samples whose reference becomes empty after this stripping are excluded from the WER aggregate, since an empty reference is unscorable and would otherwise count every emitted word as a pure insertion error. No analogous empty-reference filter is applied to the other ASR benchmarks, which do not use such tags.

Figure A shows the LLM judge configuration and prompt used for WavCaps and Alpaca-Audio, with model and decoding hyperparameters listed above the prompt box. Inside the box, **System.** and **User.** mark the chat roles, **[Section]** markers are part of the literal prompt, and *italic blue* text denotes per-sample variables. We normalized the 5-point scale scores to a 100-point scale.

C Inference Cost Analysis

Setup. Two complementary measurements support Table 2. For the main table, we run all variants on the full MMAU and MMAR evaluation sets (2,000 samples total) and report per-sample mean latency. For the duration-stratified analysis, we draw a fixed 300 samples per audio-duration range from the broader evaluation suite, which is otherwise heavily skewed toward short audio with over 90% of samples below 15 s, giving 1,500 samples spanning 0 to 120 s, and repeat the sweep with

Alpaca-Audio & WavCaps (single response, 0–100)

Judge model: `gpt-4o-2024-11-20` | *Decoding:* temperature 0, max tokens 1024 | *System message:* none | *Variables:* *{question}*, *{ground_truth}*, *{model_response}*.

LLM Judge prompt for Alpaca-Audio & WavCaps

User.
[Question]
{question}
[Reference Answer]
{ground_truth}
[Model Answer]
{model_response}
[Task]
Rate the model's answer based on its alignment with the reference answer, focusing on accuracy and relevance to the reference provided. Please be critical on the details. Criteria: Assess if the model's response mirrors the reference in terms of content, accuracy, and relevance.

Score 0: The answer is completely misaligned, providing incorrect or irrelevant information compared to the reference.

Score 1: The answer shows minimal alignment, often misunderstanding or providing irrelevant details unrelated to the reference.

Score 2: The answer recognizes the topic but diverges significantly from the reference in accuracy or relevance.

Score 3: The answer aligns with the reference generally but lacks detail or precise accuracy in some aspects.

Score 4: The answer is mostly accurate and relevant, closely following the reference but could be clearer or more detailed.

Score 5: The answer is highly accurate, detailed, and matches the reference answer perfectly, capturing its essence and detail.

Your response should be formatted as follows:
Explanation: (Provide a concise explanation of your rating, comparing the reference answer with the model's response. "The reference answer is `[XXX]`, while the model's answer is `[YYY]`. I think ...")
Rating: (int)

Figure A: LLM judge configuration and prompt for Alpaca-Audio & WavCaps.

three independent random seeds. Inference is run with batch size 1 on 4×NVIDIA B200 (tensor parallel size 4) under vLLM 0.19, with prefix caching disabled so that every request pays its full prefill cost. Note that the performance of the AF3-think model is evaluated with GPT-eval due to the answer parsing problem.

Per-duration results. Table E reports time to first token, decode time, total wall-clock, decoded tokens, and CoAT's forced-prefix length for each audio-duration range as mean ± std across 3 random seeds. Two patterns stand out. First, CoAT's forced-prefix length grows monotonically with audio duration and spans more than an order of magnitude across the five ranges. Despite this growth, CoAT's time to first token stays within a few milliseconds of the same-backbone baseline in every range, indicating that the additional prefill compute is largely absorbed by the audio encoder's existing cost. Second, text-CoT's decoded-token count stays roughly flat across all five ranges, reflecting that the textual chain-of-thought is determined by task format rather than audio duration. Its decode wall-clock is correspondingly stable across ranges and contributes the dominant cost regardless of input length.

D Inference-Time Think Token vs. CoAT

A natural alternative to CoAT is to ask whether the gains we attribute to our reconstruction teachers can in fact be obtained for free, simply by allocating extra inference-time compute to a think segment. On the Audio Flamingo 3 backbone we isolate this question by comparing two settings: an inference-time think policy that prepends a think segment to the frozen pretrained model, and our CoAT fine-tune that supervises the same segment with multi-teacher reconstruction targets. Table F reports per-benchmark scores together with the gap Δ between CoAT and another model.

Table E: **Inference cost by audio duration.** Per-duration mean ± std across 3 random seeds, with 300 samples per range per seed. TTFT, decode, and total are reported in seconds. Forced prefix is the length in tokens of CoAT's prepended thinking block. "–" marks variants that do not prepend a thinking block. Hardware and serving configuration follow Appendix C.

Duration	Model	TTFT ↓	Decode ↓	Total ↓	Dec. tok ↓	Forced prefix
0–5 s	Qwen2.5-Omni	0.136 ± 0.004	0.036 ± 0.004	0.172 ± 0.001	14.9 ± 1.7	–
	w/ text-CoT	0.142 ± 0.009	0.075 ± 0.007	0.217 ± 0.003	30.5 ± 2.8	–
	w/ CoAT	0.147 ± 0.010	0.023 ± 0.002	0.170 ± 0.012	9.8 ± 0.8	71 ± 1
	Audio Flamingo 3	0.029 ± 0.000	0.015 ± 0.001	0.044 ± 0.000	6.5 ± 0.3	–
	w/ text-CoT	0.031 ± 0.004	0.151 ± 0.008	0.183 ± 0.010	59.8 ± 3.3	–
	w/ CoAT	0.029 ± 0.000	0.021 ± 0.001	0.051 ± 0.001	9.3 ± 0.4	71 ± 1
5–15 s	Qwen2.5-Omni	0.131 ± 0.005	0.070 ± 0.006	0.201 ± 0.009	28.9 ± 2.2	–
	w/ text-CoT	0.139 ± 0.006	0.091 ± 0.002	0.230 ± 0.007	37.1 ± 0.6	–
	w/ CoAT	0.134 ± 0.004	0.063 ± 0.001	0.198 ± 0.004	26.1 ± 0.4	220 ± 4
	Audio Flamingo 3	0.029 ± 0.000	0.054 ± 0.003	0.083 ± 0.003	22.0 ± 1.1	–
	w/ text-CoT	0.029 ± 0.001	0.176 ± 0.004	0.206 ± 0.004	70.0 ± 1.3	–
	w/ CoAT	0.033 ± 0.000	0.062 ± 0.003	0.095 ± 0.003	25.3 ± 1.2	220 ± 4
15–30 s	Qwen2.5-Omni	0.134 ± 0.004	0.121 ± 0.008	0.255 ± 0.007	49.2 ± 3.3	–
	w/ text-CoT	0.142 ± 0.005	0.164 ± 0.005	0.306 ± 0.006	66.2 ± 1.9	–
	w/ CoAT	0.134 ± 0.008	0.046 ± 0.007	0.180 ± 0.002	18.9 ± 2.8	557 ± 2
	Audio Flamingo 3	0.035 ± 0.001	0.023 ± 0.007	0.058 ± 0.006	9.6 ± 2.6	–
	w/ text-CoT	0.036 ± 0.001	0.183 ± 0.002	0.218 ± 0.003	72.1 ± 0.8	–
	w/ CoAT	0.038 ± 0.000	0.039 ± 0.012	0.077 ± 0.012	15.9 ± 4.6	557 ± 2
30–60 s	Qwen2.5-Omni	0.142 ± 0.007	0.043 ± 0.005	0.185 ± 0.012	17.8 ± 2.0	–
	w/ text-CoT	0.150 ± 0.009	0.175 ± 0.006	0.325 ± 0.013	70.5 ± 2.2	–
	w/ CoAT	0.143 ± 0.009	0.036 ± 0.004	0.179 ± 0.013	14.9 ± 1.6	839 ± 3
	Audio Flamingo 3	0.048 ± 0.001	0.014 ± 0.003	0.062 ± 0.004	6.2 ± 1.3	–
	w/ text-CoT	0.048 ± 0.001	0.157 ± 0.006	0.205 ± 0.007	61.9 ± 2.3	–
	w/ CoAT	0.050 ± 0.000	0.044 ± 0.004	0.093 ± 0.004	17.8 ± 1.4	839 ± 3
60–120 s	Qwen2.5-Omni	0.183 ± 0.007	0.017 ± 0.001	0.201 ± 0.007	7.7 ± 0.3	–
	w/ text-CoT	0.160 ± 0.010	0.128 ± 0.003	0.288 ± 0.011	52.3 ± 1.0	–
	w/ CoAT	0.193 ± 0.009	0.012 ± 0.000	0.205 ± 0.009	5.6 ± 0.0	2034 ± 62
	Audio Flamingo 3	0.075 ± 0.001	0.004 ± 0.001	0.079 ± 0.001	2.2 ± 0.2	–
	w/ text-CoT	0.074 ± 0.002	0.142 ± 0.002	0.217 ± 0.002	55.8 ± 0.6	–
	w/ CoAT	0.083 ± 0.002	0.021 ± 0.005	0.104 ± 0.003	8.9 ± 1.8	2034 ± 62

The Audio Flamingo 3 denotes the vanilla AF3 model, whereas "think" or "with CoT" refers the AF3 with additional thinking. The two policies behave very differently. Inference-time thinking by itself is largely neutral or harmful, regressing on a clear majority of benchmarks across audio understanding, music, and emotion families, with the largest drops concentrated on captioning and open-ended QA. CoAT, in contrast, delivers consistent gains on the metrics that exercise audio understanding—most prominently in voice-assistant style QA, AIR-Bench Foundation, and speech emotion recognition—at the cost of a modest WER regression on most ASR test sets, consistent with mild distribution shift from fine-tuning on a multi-task corpus. The takeaway is that the benefit of CoAT comes from what the think segment is supervised to reconstruct, not from the segment's mere presence at inference.

E Societal Impact

CoAT improves audio understanding in audio language models, with potential positive applications such as accessibility (captioning, transcription, assistive listening) and content moderation. As with audio language models in general, the same improvements may have dual-use implications that warrant consideration at deployment time. These considerations apply at the level of the underlying backbone rather than being introduced by CoAT itself.

Table F: **AF3 think-token comparison.** Per-benchmark performance of Audio Flamingo 3 with the inference-time + think token policy and our + CoAT fine-tune. Since AF3 think model does not support adaptive thinking mode, we cannot evaluate ASR with WER and therefore omit this value.

Benchmark	Eval	Audio Flamingo 3	+ think	+ CoAT	Δ
Audio Understanding & Reasoning					
General					
MMAU	Acc ↑	69.40	64.52	**70.00**	+0.60 / +5.48
MMAR	Acc ↑	55.70	54.25	**59.60**	+3.90 / +5.35
MMSU	Acc ↑	**60.01**	57.03	58.36	−1.65 / +1.33
ClothoAQA	Acc ↑	80.10	68.79	**85.30**	+5.20 / +16.51
Alpaca-Audio	GPT ↑	38.80	30.10	**58.59**	+19.79 / +28.49
WavCaps	GPT ↑	**33.40**	41.00	29.00	−4.40 / −12.00
AIR-Bench Foundation					
ABF Speech	Acc ↑	62.99	58.04	**71.24**	+8.25 / +13.20
ABF Sound	Acc ↑	65.56	63.39	**69.50**	+3.94 / +6.11
ABF Music	Acc ↑	58.62	54.40	**64.50**	+5.88 / +10.10
Music Classification					
VocalSound	Acc ↑	**93.06**	82.22	92.39	−0.67 / +10.17
GTZAN	Acc ↑	94.99	92.89	**95.50**	+0.51 / +2.61
MuchoMusic	Acc ↑	81.63	75.76	**81.80**	+0.17 / +6.04
Speech Emotion Recognition					
MELD	ACC ↑	40.75	40.46	**59.83**	+19.08 / +19.47
MELD	F1 ↑	45.93	44.96	**57.16**	+11.23 / +12.20
IEMOCAP	Acc ↑	63.58	56.98	**70.39**	+6.81 / +13.41
Automatic Speech Recognition					
LibriSpeech-clean	WER ↓	**1.57**	-	1.99	+0.42 / -
LibriSpeech-other		**3.13**	-	4.23	+1.10 / -
Common Voice 15		**7.40**	-	**7.40**	+0.00 / -
GigaSpeech		**10.27**	-	11.90	+1.63 / -
VoxPopuli		**5.55**	-	5.70	+0.15 / -
SPGISpeech		1.86	-	**1.84**	−0.02 / -
Switchboard		8.01	-	**7.18**	−0.83 / -

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes], [No], or [N/A].
- [N/A] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for [N/A]).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will also be asked to include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While [Yes] is generally preferable to [No], it is perfectly acceptable to answer [No] provided a proper justification is given (e.g., error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering [No] or [N/A] is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The contributions stated in the abstract and Section 1 match the method in Section 3 and the experimental results in Section 4.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

Justification: Limitations and future work are discussed in the last part of the main submission.

Guidelines:

- The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The contribution is a methodological framework supported by controlled experiments and ablations in Section 4, and the paper does not state formal theorems that would require accompanying assumptions or proofs.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

Justification: Architecture, training schedule, and hyperparameters are described in Section 3, Section 4.1, and the appendix.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: Code and trained checkpoints will be released upon acceptance.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.

- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Optimizer, schedule, hyperparameters, and dataset details are reported in Section 4.1 and Appendix A.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: We report single-run results due to the cost of training and evaluating across three backbones on a broad benchmark suite.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

Justification: GPU type and total compute (B200 GPU-hours) per backbone and per evaluation are reported in Section 4.1 and Appendix A.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: The research conforms with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Positive and negative societal impacts are discussed in Appendix E.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: We do not release new datasets or models with high risk of misuse.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: All backbones, expert encoders, training datasets, and evaluation benchmarks are cited with their original sources in Section 4.1 and the appendix.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [N/A]

Justification: No new datasets or pretrained models are released with the submission.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.

- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The work involves no crowdsourcing or human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The work involves no human subjects research.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs were used only for writing editing, and formatting purposes, and did not contribute to the core methodology.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.