AURALLOOP: LANGUAGE-GUIDED ENCODER RE-ENTRY FOR AUDIO QUESTION ANSWERING

Anonymous authors
Paper under double-blind review

ABSTRACT

One-pass audio-language models complete acoustic encoding before language-side inference, so intermediate language states cannot guide further encoder computation. We introduce AURALLOOP, a parameter-efficient architecture that makes acoustic encoding responsive to evolving language context. A shared low-rank cell conditions a cached acoustic anchor on an intermediate language state and reruns the original frozen encoder tail. Incremental embedding differences are injected into existing audio-token positions at later language layers. Two re-entry steps share the anchor and encoder weights, with the second conditioned on a language state that has incorporated the first revision. Instantiated on Qwen2.5-Omni-7B, AURALLOOP uses 3.20M trainable parameters while keeping pretrained backbone weights frozen. It requires no additional encoder, generated control tokens, or chain-of-thought supervision during adaptation. Evaluations on ADQA, VoxParadox, MMSU, and MUGEN compare re-entry with direct acoustic adaptation, static feature access, and additional training computation. On VoxParadox, AURALLOOP improves mean accuracy by 1.10 percentage points over a single re-entry at a later language layer and by 0.97 points over joint audio-language LoRA under a controlled training-work budget. Same-depth interventions suggest task-dependent benefits from updated language conditioning, while capability analyses show the largest gains in paralinguistic perception. Paired error and latency analyses characterize the accompanying answer regressions and prefill cost. These results support language-guided encoder re-entry as a targeted approach to acoustic question answering.

1 INTRODUCTION

An answer about speech can depend on how something is said, not just on the words. Intonation can contradict a lexical claim, and distinguishing speakers may require comparing several recordings. Large audio-language models provide a general interface to these tasks, yet benchmarks reveal persistent gaps in fine-grained acoustic understanding (Pang et al., 2026; Wang et al., 2026; Yang et al., 2026b). The relevant evidence also depends on the question. The same recording may require attention to vocal delivery, speaker identity, or spoken content. These demands raise an architectural question: *can intermediate language context guide an additional encoder computation toward acoustic distinctions relevant to the question?*

In a conventional pipeline, the acoustic encoder and projector finish processing the audio before language inference begins. Audio-token states continue to evolve inside the language model, but the acoustic encoder cannot respond to that evolving context. Intermediate-feature access, question-conditioned mixing, and encoder adaptation improve how acoustic information reaches the language model (Yang et al., 2026a; Pang et al., 2026; Hu et al., 2022). They motivate a further step: use the language state to transform an earlier acoustic representation, then let subsequent language layers process the revision. Figure 1 illustrates this return path.

AURALLOOP implements this interaction with a cached acoustic anchor for each clip and one shared low-rank cell. Each call modulates the anchor, reruns the frozen encoder tail, and injects an incremental difference into existing audio-token positions. The second call uses a language state incorporating the first revision. The model learns this depth-unrolled interaction from final answers without generated control requests or rationale supervision.

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

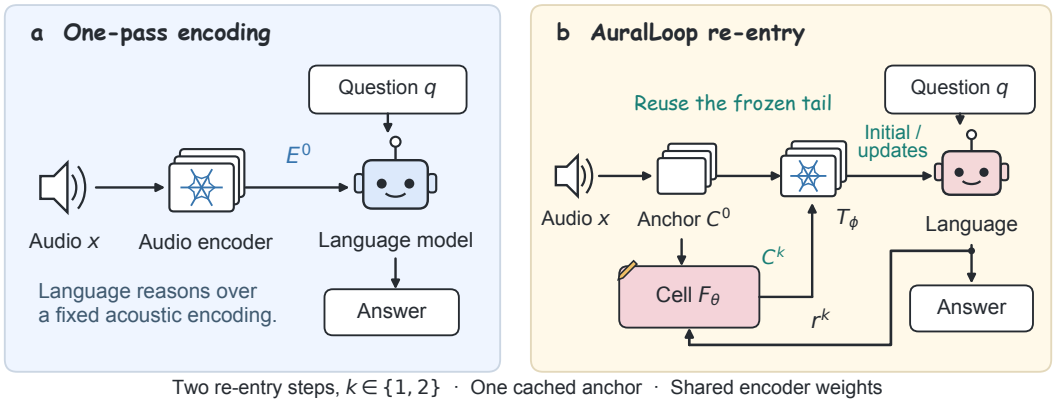

Figure 1: **From one-pass encoding to language-guided re-entry.** Both architectures process the question and evolve audio-token states within the language model. AURALLOOP adds a return path to the original frozen encoder tail. A later language state incorporates the first revision before conditioning the second call. The overview abbreviates the difference, scaling, and injection operations detailed in Figure 2.

Direct acoustic adaptation might already expose the relevant cues, and a single late call might provide sufficient context. Additional training or computation could also improve accuracy without information from the first revision. Shared-backbone comparisons and a same-depth shadow condition test these alternatives. The shadow constructs the second condition without the first injection, which remains active in the main inference path. This distinguishes sensitivity to the second update from sensitivity to its refreshed condition.

Our contributions are threefold. First, we introduce a shared encoder re-entry mechanism that couples intermediate language states to renewed acoustic computation with 3.20M trainable parameters. Both calls reuse the cached anchor and frozen tail, while the second condition incorporates the first acoustic revision.

Second, we evaluate this interaction against direct adaptation, static feature access, a late single call, and additional training work on four audio-QA benchmarks. Same-depth interventions test sensitivity to language context updated by the first revision.

Third, we characterize the resulting capability and cost tradeoff through acoustic subcategories, paired correctness changes, and prefill latency. These analyses connect the aggregate gains to paralinguistic perception and quantify the accompanying regressions and computational cost.

2 RELATED WORK

Access to acoustic representations. PAL combines compact audio summaries at the language-model input with attention-based access to the full audio sequence at selected layers, reducing the cost of exposing existing acoustic evidence (Alex et al., 2025). PCLM selects a prompt-dependent mixture of audio-encoder layers, addressing the loss of paralinguistic cues with depth. Its full system also uses preference optimization to favor acoustically supported answers (Pang et al., 2026). MOSS-Audio uses DeepStack to inject features from multiple encoder depths and time markers to preserve temporal cues, providing another route to richer acoustic access (Yang et al., 2026a). These interfaces motivate our feature-access controls. AURALLOOP computes language-conditioned tail revisions: an intermediate language state changes the input to a new encoder-tail computation. Our PCLM and DeepStack ports test feature mixing and injection under the shared adaptation protocol. Released systems are compared separately.

Adaptation and inference interventions. Omni-R1 applies group relative policy optimization (GRPO) to audio QA and finds gains from text-only adaptation, motivating a distinction between language-side improvements and better perception (Rouditchenko et al., 2025). Audio-Aware Decoding contrasts token logits with and without audio to favor audio-supported predictions, offering an

inference-time alternative to changing the representation pathway (Hsu et al., 2025). IAAN contrasts encoder-neuron activations on the waveform and a noise reference, then amplifies selected neurons without training. It directly tests the value of intervening inside the acoustic encoder (Huang et al., 2026). We compare both inference methods on the same language-LoRA checkpoints and include joint audio-language adaptation with additional training work. Together, these controls test whether re-entry contributes beyond stronger adaptation or inference-time amplification.

Language-guided re-perception. Echo learns to locate and revisit informative waveform segments through supervised training and reinforcement learning, interleaving renewed audio access with generated reasoning (Wu et al., 2026a). It motivates revisiting acoustic evidence, whereas AURALLOOP revises a cached representation during prompt prefill. In vision, Visual Perception Tokens use a generated token's hidden state to guide visual re-encoding, establishing a close precedent for language-conditioned encoder computation (Yu et al., 2025). ViLaVT uses language-guided modulation and a dynamic vision encoder to jointly re-encode image regions, coupling visual updates to linguistic reasoning (Wu et al., 2026b). AURALLOOP instantiates this interaction through fixed language-layer taps, a shared low-rank cell, and the original frozen audio tail. Both calls restart from each clip's acoustic anchor and return incremental updates to existing token positions. The second condition incorporates the first revision without generating a re-perception request. Appendix A expands these comparisons.

3 AURALLOOP

3.1 AN ACOUSTIC ANCHOR INSIDE A STAGED LANGUAGE COMPUTATION

AURALLOOP interleaves language processing with two revisions of a cached acoustic representation (Figure 2). The audio stem runs once. The original encoder tail runs initially and once at each re-entry. Each language stage advances the complete token sequence and provides a prompt-end conditioning state as a side output.

Let $x = (x_1, \ldots, x_m)$ contain m ordered audio clips and q be the complete observed prompt, including options. The target answer $y = (y_1, \ldots, y_{T_y})$ has length T_y, including its native end token. For clip i, the stem S_ψ and tail T_ϕ, with frozen parameters ψ and ϕ, produce

$$C_i^0 = S_\psi(x_i; \mu_i) \in \mathbb{R}^{T_{c,i} \times d_a}, \qquad E_i^0 = T_\phi(C_i^0; \mu_i) \in \mathbb{R}^{T_{a,i} \times d_\ell}. \tag{1}$$

Here d_a and d_ℓ are acoustic and language widths. The tail retains native output reconstruction, pooling, post-normalization, and projection. Metadata μ_i preserves valid lengths and masks, attention-chunk boundaries, positional bookkeeping, and pooling alignment. The lengths $T_{c,i}$ and $T_{a,i}$ count valid rows before and after pooling. Clips are encoded separately.

Native packing combines $\{E_i^0\}$ and q into $H^0 \in \mathbb{R}^{n \times d_\ell}$. During prefill, n counts prompt positions. Under teacher forcing, the sequence also includes appended answer tokens. The ordered indices $\mathcal{A}_i = (a_{i,1}, \ldots, a_{i,T_{a,i}})$ identify clip i's audio-token rows. Let p be the final prompt position before the answer. Matrices store positions as rows, and individual feature vectors are their transposes. Denote causal language layers a through b by $D_{a:b}^{\omega,\eta}$, with frozen weights ω and trainable LoRA factors η. At boundary b_k,

$$\bar{H}^k = D_{b_{k-1}+1:b_k}^{\omega,\eta}(H^{k-1}), \qquad r^k = (\bar{H}_{p,:}^k)^\top \in \mathbb{R}^{d_\ell}, \quad k \in \{1, 2\},\ b_0 = 0. \tag{2}$$

For Qwen2.5-Omni-7B, $d_a = 1280$ and $d_\ell = 3584$. The acoustic encoder has 32 layers and the Thinker has 28 (Qwen Team, 2025). We cache the anchor after acoustic layer 24, repeat layers 25–32, and inject after language layers $b_1 = 8$ and $b_2 = 18$.

3.2 A SHARED LANGUAGE-CONDITIONED CELL

Omitting the clip index and writing $T_c = T_{c,i}$, let $t \in \{1, \ldots, T_c\}$ index a valid acoustic row and write $c_t^0 = (C_{t,:}^0)^\top$. With $d_b = 256$, the bias-free maps are $W_a \in \mathbb{R}^{d_b \times d_a}$ and $W_r \in \mathbb{R}^{d_b \times d_\ell}$. Affine LayerNorms $\mathrm{LN}_a, \mathrm{LN}_r$ operate over channels. The projected vectors $u_t, z^k, m_t^k \in \mathbb{R}^{d_b}$ are

$$u_t = W_a\,\mathrm{LN}_a(c_t^0), \qquad z^k = W_r\,\mathrm{LN}_r(r^k), \qquad m_t^k = u_t \odot z^k. \tag{3}$$

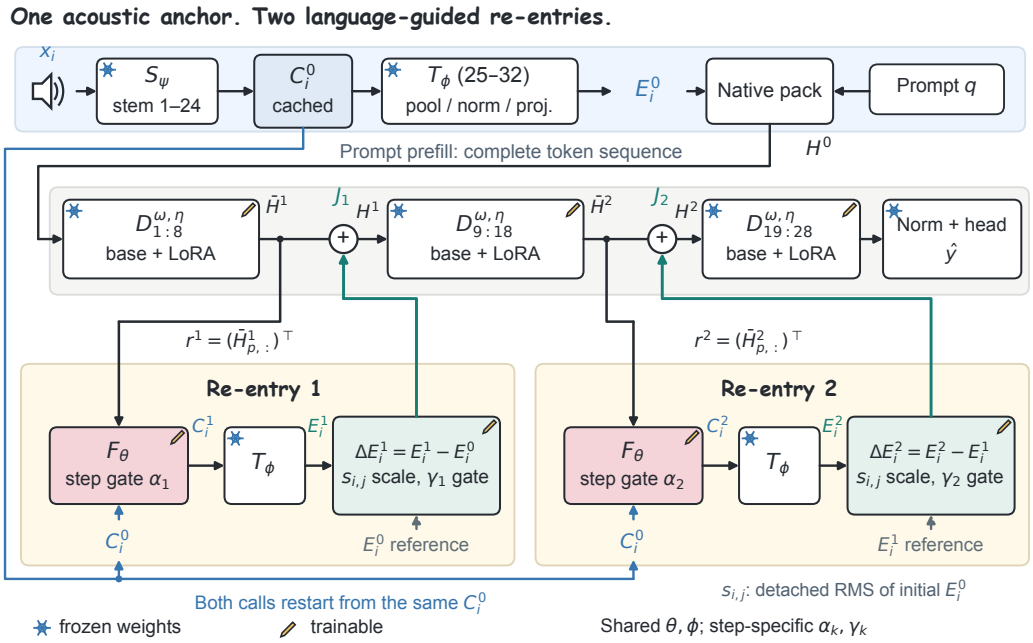

Figure 2: **The complete re-entry graph.** The full token stream advances through every language stage. Prompt-end taps r^1, r^2 condition the shared cell at C^0. All three T_ϕ calls share the frozen tail, including native post-normalization. J_k injects incremental differences at native audio slots (Equation (5)). Only the cell, language LoRA, and step gates are trained.

Here $\odot$ denotes elementwise multiplication, and z^k is broadcast across valid acoustic positions within the same example. A position-dependent scalar gate controls the update:

$$g_t^k = \sigma(w_g^\top m_t^k), \qquad c_t^k = c_t^0 + \alpha_k g_t^k W_o m_t^k, \qquad E_i^k = T_\phi(C_i^k; \mu_i). \tag{4}$$

The output map is $W_o \in \mathbb{R}^{d_a \times d_b}$, $w_g \in \mathbb{R}^{d_b}$, and σ is sigmoid. The shared parameters θ comprise these projections, both LayerNorms, and w_g. The learned scalar α_k is step-specific. We denote the complete cell by $F_\theta(C^0, r^k; \alpha_k)$ and leave padded positions unchanged.

Both calls restart from C^0 and share the cell and tail weights. The second condition r^2 incorporates the first injection through the intervening language layers.

3.3 Incremental Revisions in Native Audio-Token Slots

Re-entry preserves each clip's valid embedding length, so the revision can return to its original token positions. For $j \in \{1, \ldots, T_{a,i}\}$, define $e_{i,j}^k = (E_{i,j,:}^k)^\top$ and $\delta e_{i,j}^k = e_{i,j}^k - e_{i,j}^{k-1}$. We scale this difference using the initial embedding's regularized root-mean-square reference:

$$\begin{aligned} s_{i,j} &= \mathrm{sg}\left(\sqrt{\|e_{i,j}^0\|_2^2/d_\ell + \epsilon}\right), \\ H_{a_{i,j},:}^k &= \bar{H}_{a_{i,j},:}^k + \left(\gamma_k \odot \frac{e_{i,j}^k - e_{i,j}^{k-1}}{s_{i,j}}\right)^\top. \end{aligned} \tag{5}$$

Here $\epsilon = 10^{-6}$, sg stops gradients through the reference scale, and $\gamma_k \in \mathbb{R}^{d_\ell}$ is a learned channel gate. We denote this audio-row scatter-add by J_k. All other rows are copied from $\bar{H}^k$.

The proposals are $E^1 - E^0$ and $E^2 - E^1$. They represent successive changes in acoustic-embedding space that are processed by later language layers. The two injections do not telescope through the intervening nonlinear language computation. After the second injection, layers 19–28, the original final normalization, and the output head produce the answer distribution. Greedy decoding yields the predicted answer $\hat{y}$.

3.4 Optimization, gradients, and answer generation

All pretrained encoder, projector, language-model, and output-head weights remain frozen. On adaptation set $\mathcal{D}$, we train $\Theta = \{\theta, \eta, \alpha_1, \alpha_2, \gamma_1, \gamma_2\}$ with the conditional answer distribution p_Θ and answer-only loss:

$$\mathcal{L}(\Theta) = -\frac{1}{|\mathcal{D}|} \sum_{(x,q,y)\in\mathcal{D}} \frac{1}{T_y} \sum_{u=1}^{T_y} \log p_\Theta(y_u \mid x, q, y_{<u}). \tag{6}$$

Here y_u is target token u and $y_{<u}$ its preceding answer prefix. Loss is averaged over target tokens within each example and then over examples. Prompt and padding labels are ignored. Frozen tail weights retain derivatives with respect to their inputs. Gradients pass through the conditioning states into language LoRA and through the subtraction term $-E^1$. Differences, reference reductions, and the loss are evaluated in FP32. The exact trainable count is 3,195,650 (Appendix H.2).

Both re-entries occur during prompt prefill. With p fixed and causal masks preserved, neither tap accesses answer tokens, including under teacher forcing. Subsequent answer tokens traverse the full language stack using the corresponding layer-wise key–value (KV) caches. Appendix C establishes answer-prefix invariance and the zero-injection reduction; Appendix D specifies training and cached decoding.

4 Experimental setup

Data and evaluation. We adapt on AudioMCQ-StrongAC-GeminiCoT, a released set of 19,480 records (Harland, 2026b; He et al., 2026a). Training uses audio, questions, options, and final labels, with rationale fields discarded. After filtering and source-overlap exclusion, source-grouped training and development sets contain 17,872 and 1,440 records. Source IDs, decoded-waveform hashes, and near-duplicate matching define the overlap checks. Training excludes clips longer than 60 seconds or sequences exceeding 2,048 packed tokens. Evaluation retains complete recordings with native chunking. Figure 6 maps the full experimental protocol, and Appendix G gives preprocessing details.

We evaluate accuracy separately on ADQA-Bench (3,000 questions), VoxParadox (2,000), MMSU (5,000 across 47 tasks), and MUGEN (1,750) (Harland, 2026a; He et al., 2026b; Pang et al., 2026; Wang et al., 2026; Yang et al., 2026b). These benchmarks probe audio-dependent QA, conflicting lexical and acoustic cues, perception and reasoning, and multi-audio comparison, respectively. Additional endpoints are VoxParadox adversarial-label agreement (ALA), measuring misleading lexical-label selection, and LibriSpeech word error rate (WER) (Panayotov et al., 2015). Benchmarks are reported separately because some share source material.

Architectural comparisons. The shared-backbone track fixes adaptation data, answer-only supervision, and the evaluation interface. AuralLoop trains its cell and step gates with rank-4 Thinker query/output (q/o) LoRA. Language-only LoRA uses rank 8. Joint audio-language LoRA combines rank-20 acoustic-tail query/key/value/output (q/k/v/o) adapters with rank-4 language q/o adapters (Hu et al., 2022). The trainable comparison models are within 2% of AuralLoop's trainable-parameter count. PCLM and DeepStack are feature-access ports, while AAD and IAAN operate on the language-LoRA checkpoints. Released external systems have different training histories and appear separately in Appendix Q.

The late single-call control injects after language layer 18, and the early control after language layer 8. Joint LoRA (C) uses 710–717 updates on the same data, matching cumulative synchronized training time within 0.12% across seeds. The budget includes forward/backward computation and communication, excluding validation, saving, and CPU preprocessing. Structural ablations test projected-token feedback, a fixed question-only condition, anchor accumulation, and non-incremental injection. Fixed-checkpoint interventions further compare intact feedback with suppression, donor conditions, and a same-depth shadow condition. Appendix I defines each comparison.

Training and model selection. Equal-exposure runs use AdamW for two epochs (560 updates), global batch 64, 5% warm-up, and cosine decay. New cell, mixer, feature-interface, and gate parameters use a base learning rate 2×10^{-4}, while audio and language LoRA factors use 8×10^{-5}.

LoRA scaling is 2 and dropout is 0.05. Base dropout is disabled. The 7B runs use BF16, full parameter sharding, and activation checkpointing across eight RTX 4090 GPUs. Each GPU processes one example per microstep with eight accumulation steps.

All trained comparators share the grouped development split and learning-rate multiplier grid $\{0.5, 1, 2\}$. Checkpoints maximize development accuracy, with validation every 50 updates and at epoch ends, preferring earlier checkpoints on ties. The additional-work control preserves 13 validation opportunities at matched training-work progress. All reported configurations selected multiplier 1 and their final permitted checkpoint. Seeds are 3407, 7189, and 12011. Appendix H supplies initialization, optimizer constants, and schedules.

Scoring and reporting. Shared-backbone models use the native chat template, original option order, greedy decoding, and a 16-token label-output cap. After native terminal-marker removal and outer-whitespace stripping, the parser accepts one case-insensitive label, alone, followed by one period, or enclosed in parentheses. A native EOS/end-of-turn token is required for completion, including at the cap. Incomplete or invalid outputs and runtime failures count as errors. MUGEN's five candidate clips remain distinct from any reference clip. Tables report three-seed means and sample standard deviations, with differences computed from unrounded means. These standard deviations measure variation across training seeds, rather than uncertainty from resampling evaluation sources. Interventions keep checkpoints fixed, while structural variants are trained separately. Latency includes initial acoustic encoding and all re-entry calls, with no anchor or KV-cache reuse across requests. Appendices J, O, and P give exact parser, statistical, and timing definitions.

5 Results and Analysis

5.1 Gains beyond direct acoustic adaptation

Table 1: **Accuracy under a shared Qwen2.5-Omni-7B adaptation protocol.** Means and sample SDs over three seeds. Scores are percentages. Higher is better.

Method	ADQA	VoxParadox	MMSU	MUGEN
Same training data and two-epoch exposure				
LoRA (LLM)	49.82 ±0.22	15.45 ±0.33	62.48 ±0.25	31.45 ±0.20
LoRA (audio+LLM)	52.70 ±0.23	31.00 ±0.33	67.16 ±0.23	34.11 ±0.26
DeepStack†	50.84 ±0.15	22.05 ±0.23	64.66 ±0.24	32.34 ±0.10
PCLM†	51.63 ±0.22	27.55 ±0.28	65.81 ±0.22	32.80 ±0.23
LoRA (LLM) + AAD	50.73 ±0.27	23.42 ±0.21	63.21 ±0.28	31.18 ±0.17
LoRA (LLM) + IAAN	51.91 ±0.20	32.62 ±0.29	65.48 ±0.21	32.70 ±0.23
AuralLoop	**54.63** ±0.19	**33.20** ±0.28	**68.72** ±0.22	**35.20** ±0.23
Additional updates under the controlled training-work budget				
Joint LoRA (C)	53.69 ±0.28	32.23 ±0.26	68.10 ±0.16	34.91 ±0.17

†: port of a published mechanism. AAD/IAAN use the same language-LoRA checkpoints and incur additional inference cost. C changes training exposure, with the same data and adapter ranks.

AuralLoop achieves the highest mean accuracy among the methods in Table 1 on all four benchmarks. Joint audio-language LoRA substantially improves on language-only adaptation, establishing a strong direct-adaptation baseline. AuralLoop surpasses this independently trained joint adapter under the same acoustic supervision, with its largest margin on VoxParadox.

The additional-work control tests whether more optimization can close this gap. Joint LoRA (C) improves over ordinary joint LoRA on every benchmark, yet AuralLoop retains a 0.97-point VoxParadox advantage and positive mean margins on the other tasks. IAAN is also competitive on VoxParadox, where it directly targets acoustic attributes. AuralLoop retains higher mean accuracy than these stronger adaptation and inference-time comparators.

PCLM and DeepStack improve performance through access to existing features, but remain below AuralLoop in the shared protocol. Together with the direct-adaptation controls, these ports favor language-responsive acoustic computation over the tested feature-routing alternatives. Appendix Q

separately reports our evaluations of released checkpoints and source-reported results under their original protocols.

5.2 Encoder access and re-entry placement

Table 2: **Encoder access and re-entry placement.** Each row is trained separately. Values are three-seed mean accuracies. "Calls/tail" denotes revision calls and repeated encoder blocks per call.

Configuration	Calls/tail	ADQA	VoxP	MMSU	MUGEN
One call, layer 8	1/8	53.60	30.60	67.82	34.46
One call, layer 18	1/8	54.07	32.10	68.31	34.78
Two calls, layers 8/18	2/8	54.63	33.20	68.72	35.20
Projected-token feedback	2/0	52.20	24.85	66.08	33.31
Fixed-question conditioning	2/8	52.98	29.42	67.36	34.02

All configurations retain the full language stack. Bold identifies the default architecture. Additional call-count, residual-design, tail-depth, and gradient controls appear in Appendix M.

Moving a single re-entry from layer 8 to layer 18 strengthens the one-call model (Table 2). Two calls retain a 1.10-point VoxParadox gain over this late control. The tested late single-call model remains below the two-call model in mean accuracy. Staging lets the first revision influence the later condition. A third call yields only a small further gain at the cost of another tail evaluation (Appendix M). These results characterize the two-call default selected through the development protocol.

Projected-token feedback retains language conditioning but bypasses the acoustic tail. Its lower accuracy favors the tested pre-tail interface under a similar parameter budget. The variants also differ in bottleneck width and repeated computation, so this comparison jointly tests their representational and computational choices. Anchor reset and incremental injection outperform their tested alternatives in mean accuracy (Appendix M).

Fixed-question conditioning falls below the default despite retaining two tail calls. It supplies a pooled text-only condition and initially produces a zero second difference when the anchor scales coincide. The following fixed-checkpoint interventions complement this trained architectural control.

5.3 Updated conditioning has task-dependent value

Table 3: **What does the second condition contribute?** Accuracy decrease from intact two-call feedback, computed from unrounded three-seed means (percentage points). Positive values favor intact feedback. Weights are fixed, with no retraining.

Second-step intervention	ADQA	VoxP	MMSU	MUGEN
Zero second injection	0.73	2.30	0.79	0.57
Norm-matched donor	1.09	2.62	1.07	0.86
Same-depth shadow	0.52	1.03	0.33	0.17
Same-depth shadow, norm-matched	0.43	0.67	0.24	0.04

The receiving first revision is fixed in every row. Norm matching acts on the injected residual after reference scaling and channel gating. Absolute scores and full definitions appear in Appendix N.

Suppressing the second injection reduces accuracy, but this intervention removes the entire trained update. Reusing r^1 additionally changes the conditioning depth and can collapse the second residual when the two anchor scales coincide. We therefore evaluate a *same-depth shadow*: the middle language stage processes $\bar{H}^1$ without the first injection to produce a layer-18 condition. That condition replaces r^2 in the second cell call, while the receiving path retains its intact first revision and subtraction reference E^1.

Table 3 shows task-dependent sensitivity to replacing the refreshed condition. On VoxParadox, a norm-matched shadow loses 0.67 points, while removing the entire second injection has a larger effect. On MUGEN, the corresponding shadow difference is only 0.04 points. Sensitivity to updated conditioning is thus more task-dependent than sensitivity to the second path itself. Matching depth and residual amplitude helps distinguish these two empirical effects.

Donor conditions cause larger losses than the same-sample shadow. They perturb the relationship between the question, acoustic anchor, and conditioning state. The norm-matched shadow controls depth and whole-clip residual magnitude, while changing the conditioning distribution. Together, these controls support task-dependent use of updated language context.

5.4 GAINS CONCENTRATE IN ACOUSTIC PERCEPTION

Table 4: **MMSU capability breakdown.** Three-seed mean accuracy. Bold marks the highest value in each column.

	Perception			Reasoning			
Method	Sem.	Phon.	Para.	Sem.	Phon.	Para.	All
LoRA (LLM)	56.59	40.46	43.30	88.36	82.05	50.25	62.48
PCLM†	59.37	46.95	50.40	88.51	82.67	52.94	65.81
LoRA (audio+LLM)	59.58	49.45	53.63	**88.60**	**82.74**	55.42	67.16
AuralLoop	**60.47**	**51.16**	**59.74**	88.33	82.40	**55.72**	**68.72**

Sem.: semantic. Phon.: phonological. Para.: paralinguistic. Unrounded counts are weighted by category size. The task-specific results are not independent replications.

The MMSU breakdown localizes the improvement (Table 4). Relative to joint audio-language LoRA, paralinguistic perception improves by 6.11 points. Semantic and phonological perception also improve, while the weighted reasoning aggregate declines slightly. The gain is concentrated in perceptual categories, consistent with an intervention that renews acoustic computation. These category scores identify the performance profile rather than changes in a measured latent representation.

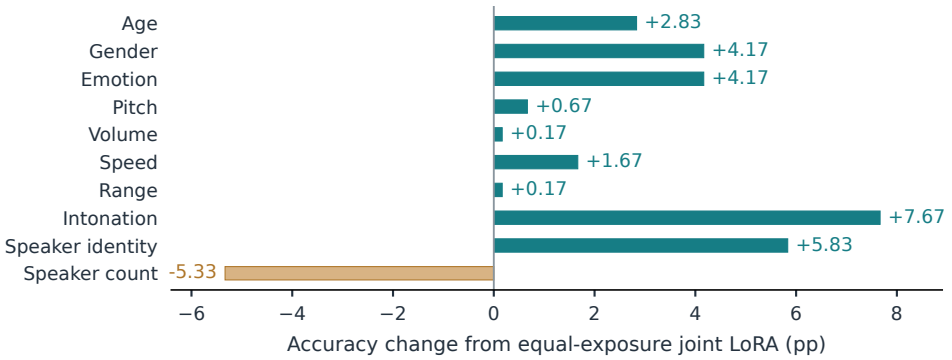

Figure 3: **Which VoxParadox attributes improve?** Three-seed mean accuracy differences from equal-exposure joint audio-language LoRA (Table 18). Each attribute contains 200 questions. Positive values favor AURALLOOP.

The attribute-level pattern is similarly selective (Figure 3). Intonation and speaker identity show the largest gains, while speaker counting declines. MUGEN temporal awareness also favors the direct-adaptation baseline (Appendix L). These breakdowns distinguish gains in acoustic delivery and identity from the remaining counting and temporal-awareness weaknesses.

On VoxParadox, ALA falls more than accuracy rises (Appendix L). The correct-label share rises, but so does the share of other errors. This three-way accounting distinguishes reduced lexical-label selection from improved correctness. These marginal proportions describe response allocation, while the paired analysis below tracks changes in individual correctness.

5.5 NET ANSWER IMPROVEMENTS AND THEIR COMPUTATIONAL COST

Figure 4 compares paired correctness between independently trained late single-call and two-call models.

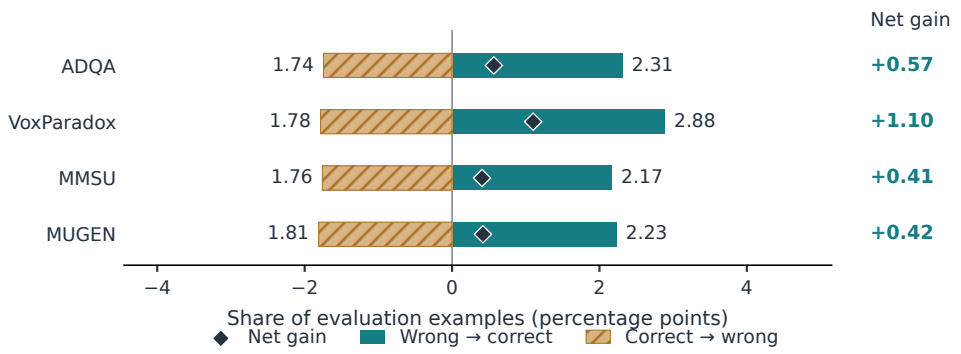

Figure 4: **Paired correctness differences between late single-call and two-call models.** Left and right bars show correct-to-wrong and wrong-to-correct rates. Labels give magnitudes, and diamonds mark their signed difference. Rates average paired counts across three seeds. The models are trained separately.

On VoxParadox, repairs exceed regressions by 1.10 percentage points, computed from unrounded counts. MUGEN's closer balance shows why net accuracy alone understates changes in individual correctness.

Appendix J.4 illustrates these outcomes with acoustic questions. On the same recording, both models answer a spoken-content question correctly, but only the two-call model resolves a conflicting pitch-contour question. A speaker-counting example reverses the comparison, with only the late-call model correct. These cases distinguish resolving lexical–acoustic conflict from enumerating sound sources. The paired counts quantify the balance of improvements and regressions across the full evaluation sets.

Two calls increase mean time to first token (TTFT) by 8.4–10.0% over the late single-call model on the QA workloads (Appendix P, Table 28). Joint LoRA (C) incurs a smaller deployment cost because its additional work occurs during training. Weight sharing limits the trainable-parameter budget, but each tail invocation adds inference computation.

Relative TTFT overhead also increases across the tested fixed-duration workloads. The QA gains coexist with a small LibriSpeech WER regression (Appendix P). Whether the remaining QA gain warrants additional prefill work depends on the application's error costs and latency budget.

Scope. The reported comparisons use Qwen2.5-Omni-7B with complete conditioning prompts. Anchor information and shared benchmark sources constrain the scope of this evidence (Appendix R).

6 Conclusion

AuralLoop makes acoustic encoding responsive to language context that evolves during the same prompt-prefill computation. A shared low-rank cell revisits a cached anchor through the original frozen encoder tail, and incremental updates return the revised evidence to existing audio-token positions. The second call uses a language state that has incorporated the first revision, completing the two-step interaction.

Across the four evaluated audio-QA benchmarks, AuralLoop improves mean accuracy over direct adaptation, static feature access, and a late single call. The training-work control and same-depth interventions clarify the value of renewed acoustic computation and its task-dependent sensitivity to refreshed conditioning. Capability and paired-error analyses locate the strongest gains in paralinguistic perception while exposing regressions and prefill cost. These findings support a targeted design principle: intermediate language context can guide computation within a pretrained acoustic encoder to improve selected acoustic judgments. The architecture realizes this interaction without additional encoder weights or generated control tokens.

REPRODUCIBILITY STATEMENT

Section 3 specifies the complete forward computation and training objective. Appendices B–E provide tensor conventions, formal properties, pseudocode, and initialization. Appendices G–J describe data processing, adaptation settings, baselines, and scoring. The result appendices retain seed-level counts and definitions of paired comparisons, interventions, and timing workloads. The separate single-device reference code illustrates the model interfaces. Its package does not include the distributed-training, training-work measurement, or deployment-profiling drivers.

ETHICS STATEMENT

This work studies existing audio datasets and pretrained models. Speech may contain personal information and copyrighted material; any redistribution must respect the licenses and access conditions of the original sources. Labels for perceived age, gender, identity, and emotion describe benchmark annotations, rather than verified personal characteristics or internal states. The method is intended for audio understanding research, and its performance does not justify consequential inferences about individual speakers.

AI USE STATEMENT

Generative AI assisted methodological development, literature review, result interpretation, mathematical and implementation checks, and manuscript preparation. The authors are responsible for all scientific claims, empirical results, and AI-assisted content (ICLR, 2026b;a).

REFERENCES

Tony Alex, Wish Suharitdamrong, Sara Atito, Armin Mustafa, Philip J. B. Jackson, Imran Razzak, and Muhammad Awais. PAL: Probing audio encoders via LLMs – audio information transfer into LLMs. *arXiv preprint arXiv:2506.10423*, 2025. URL https://arxiv.org/abs/2506.10423v3. Revised February 2026.

Santiago Cuervo, Skyler Seto, Maureen de Seyssel, Richard Bai, Zijin Gu, Tatiana Likhomanenko, Navdeep Jaitly, and Zakaria Aldeneh. Closing the gap between text and speech understanding in LLMs. In *International Conference on Learning Representations*, pp. 146313–146335, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/ec4c05921d70202023898568a2a8c002-Abstract-Conference.html.

DCASE Task 5 Organizers. DCASE 2026 task 5: Audio-dependent question answering results, 2026. URL https://dcase.community/challenge2026/task-audio-dependent-question-answering-results. Accessed 24 September 2026.

Harland. ADQA-Bench: Audio-dependent question answering evaluation benchmark, 2026a. URL https://huggingface.co/datasets/Harland/ADQA-Bench. Accessed 24 September 2026.

Harland. AudioMCQ-StrongAC-GeminiCoT: Dataset card, 2026b. URL https://huggingface.co/datasets/Harland/AudioMCQ-StrongAC-GeminiCoT. Accessed 24 September 2026.

Haolin He, Xingjian Du, Renhe Sun, Zheqi Dai, Yujia Xiao, Mingru Yang, Jiayi Zhou, Xiquan Li, Zhengxi Liu, Zining Liang, Chunyat Wu, Qianhua He, Tan Lee, Xie Chen, Wei-Long Zheng, Weiqiang Wang, Mark D. Plumbley, Jian Liu, and Qiuqiang Kong. Measuring audio's impact on correctness: Audio-contribution-aware post-training of large audio language models. In *International Conference on Learning Representations*, pp. 110411–110432, 2026a. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/b2fbf1c9bc92e7ef2f6cab2e8a3e09af-Abstract-Conference.html.

Haolin He, Renhe Sun, Zheqi Dai, Xingjian Du, Chunyat Wu, Zining Liang, Zhengxi Liu, Jiahe Lei, Runbang Wang, Jiayi Zhou, Mingru Yang, Xiquan Li, Yun Chen, Xie Chen, Zhiyao Duan, Weiqiang Wang, Mark D. Plumbley, Jian Liu, and Qiuqiang Kong. Summary of DCASE 2026 task 5: Audio-dependent question answering. *arXiv preprint arXiv:2607.18718*, 2026b. URL https://arxiv.org/abs/2607.18718.

Tzu-Wen Hsu, Ke-Han Lu, Cheng-Han Chiang, and Hung-yi Lee. Reducing object hallucination in large audio-language models via audio-aware decoding. In *IEEE Automatic Speech Recognition and Understanding Workshop*, pp. 1–7, 2025. doi: 10.1109/asru65441.2025.11434595. URL https://arxiv.org/abs/2506.07233.

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. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Yu-Han Huang, Chih-Kai Yang, Ke-Han Lu, An-Yu Cheng, and Hung-yi Lee. Encoder-side neuron identification and amplification for acoustic perception in large audio-language models. *arXiv preprint arXiv:2607.11801*, 2026. URL https://arxiv.org/abs/2607.11801.

ICLR. ICLR 2027 AI policy for authors, 2026a. URL https://iclr.cc/Conferences/2027/AIPolicyForAuthors. Accessed 24 September 2026.

ICLR. ICLR 2027 author guidelines, 2026b. URL https://iclr.cc/Conferences/2027/AuthorGuidelines. Accessed 24 September 2026.

Tzu-Quan Lin, Wei-Ping Huang, Yi-Cheng Lin, and Hung-yi Lee. How contrastive decoding enhances large audio language models. *arXiv preprint arXiv:2603.09232*, 2026. URL https://arxiv.org/abs/2603.09232. Accepted at IEEE Spoken Language Technology Workshop (SLT) 2026.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. LibriSpeech: An ASR corpus based on public domain audio books. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 5206–5210, 2015. doi: 10.1109/ICASSP.2015.7178964.

Jiacheng Pang, Ashutosh Chaubey, and Mohammad Soleymani. Do audio LLMs listen or read? analyzing and mitigating paralinguistic failures with VoxParadox. In *International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2605.27772.

Qwen Team. Qwen2.5-Omni-7B: Model configuration, 2025. URL https://huggingface.co/Qwen/Qwen2.5-Omni-7B/blob/main/config.json. Accessed 24 September 2026.

Andrew Rouditchenko, Saurabhchand Bhati, Edson Araujo, Samuel Thomas, Hilde Kuehne, Rogerio Feris, and James Glass. Omni-R1: Do you really need audio to fine-tune your audio LLM? In *IEEE Automatic Speech Recognition and Understanding Workshop*, pp. 1–7, 2025. doi: 10.1109/asru65441.2025.11434780. URL https://arxiv.org/abs/2505.09439.

Yi-Jen Shih, Desh Raj, Chunyang Wu, Wei Zhou, SK Bong, Yashesh Gaur, Jay Mahadeokar, Ozlem Kalinli, and Mike Seltzer. Can speech LLMs think while listening? In *International Conference on Learning Representations*, pp. 72392–72414, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/75c45fca2aa416ada062b26cc4fb7641-Abstract-Conference.html.

Vishaal Udandarao, Zhiyun Lu, Xuankai Chang, Yongqiang Wang, Albin Madappally Jose, Fartash Faghri, Josh Gardner, and Chung-Cheng Chiu. Data-centric lessons to improve speech-language pretraining. In *International Conference on Learning Representations*, pp. 61136–61190, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/63b96ace3e28465aff61918e77de2a00-Abstract-Conference.html.

Dingdong Wang, Junan Li, Jincenzi Wu, Dongchao Yang, Xueyuan Chen, Tianhua Zhang, and Helen Meng. MMSU: A massive multi-task spoken language understanding and reasoning benchmark. In *International Conference on Learning Representations*, pp. 31374–31410,

2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/34e278fbbd7d6d7d788c98065988e1a9-Abstract-Conference.html.

Daiqing Wu, Xuan Zhang, Dongbao Yang, Jiashu Yao, Longfei Chen, Qingsong Liu, Sicheng Zhao, Can Ma, Yangyang Kang, and Yu Zhou. Echo: Towards advanced audio comprehension via audio-interleaved reasoning. In *International Conference on Learning Representations*, pp. 116975–117004, 2026a. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/bde5704ac61254698db5fcebf3a33f41-Abstract-Conference.html.

Junfei Wu, Jian Guan, Qiang Liu, Shu Wu, Liang Wang, Wei Wu, and Tieniu Tan. Chatting with images for introspective visual thinking. *arXiv preprint arXiv:2602.11073*, 2026b. URL https://arxiv.org/abs/2602.11073.

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. URL https://arxiv.org/abs/2503.20215.

Chen Yang, Chufan Yu, Hanfu Chen, Jie Zhu, Jingqi Chen, Ke Chen, Wenxuan Wang, Yang Wang, Yaozhou Jiang, Yi Jiang, Zhengyuan Lin, Ziqi Chen, Zhaoye Fei, Chenghao Liu, Donghua Yu, Jun Zhan, Kang Yu, Kexin Huang, Liwei Fan, Mingshu Chen, Qinyuan Cheng, Ruixiao Li, Shimin Li, Songlin Wang, Xingjian Zhao, Yang Gao, Yitian Gong, Yiyang Zhang, Zhe Xu, and Xipeng Qiu. MOSS-Audio technical report. *arXiv preprint arXiv:2606.01802*, 2026a. URL https://arxiv.org/abs/2606.01802.

Chih-Kai Yang, Yun-Shao Tsai, Yu-Kai Guo, Ping-Le Tsai, Yen-Ting Piao, Hung-Wei Chen, Ting-Lin Hsiao, Yun-Man Hsu, Ke-Han Lu, and Hung-yi Lee. MUGEN: Evaluating and improving multi-audio understanding of large audio-language models. In *Interspeech*, pp. 3803–3808, 2026b. doi: 10.21437/Interspeech.2026-530. URL https://www.isca-archive.org/interspeech_2026/yang26c_interspeech.html.

Runpeng Yu, Xinyin Ma, and Xinchao Wang. Auto-controlled image perception in MLLMs via visual perception tokens. In *IEEE/CVF International Conference on Computer Vision*, pp. 21822–21831, 2025. URL https://openaccess.thecvf.com/content/ICCV2025/html/Yu_Auto-Controlled_Image_Perception_in_MLLMs_via_Visual_Perception_Tokens_ICCV_2025_paper.html.

APPENDIX

The appendix provides related-work context, mathematical and implementation details, the shared experimental protocol, reported seed-level counts, and additional analyses of feedback, errors, and computational cost.

A EXTENDED RELATED WORK

Three lines of research motivate the controlled comparisons: improving access to existing acoustic representations, adapting acoustic computation, and revisiting perceptual inputs during inference.

Access to acoustic representations. PAL combines compact input summaries with attention-based access to the full audio sequence (Alex et al., 2025). PCLM mixes encoder depths according to the question, with optional preference optimization (Pang et al., 2026). MOSS-Audio combines DeepStack with time markers and extensive training (Yang et al., 2026a). These approaches improve the audio–language interface without conditioning a second encoder-tail transformation on the language state produced after a first revision. Our PCLM and DeepStack ports retain their respective mixing and injection mechanisms under the shared adaptation budget. The released complete systems remain a separate comparison because their training histories differ.

Active perception. Echo uses generated segment references to initiate re-listening, supported by localization-aware supervised training and reinforcement learning (Wu et al., 2026a). Visual Perception Tokens use the hidden state of a generated token to guide visual re-encoding (Yu et al., 2025). ViLaVT combines language-guided modulation with a dynamic visual encoder (Wu et al., 2026b). AuralLoop instead places latent conditions at fixed prefill depths, retaining one acoustic anchor per clip and the original frozen tail without generating a perceptual request. Its contribution is this parameter-efficient computation graph and its controlled evaluation, rather than the general idea of top-down or hidden-state-guided perception.

Inference interventions. AAD contrasts audio-conditioned and reference predictions, while IAAN intervenes on encoder neurons (Hsu et al., 2025; Huang et al., 2026). Both can improve answers without the proposed adaptation architecture, making them relevant alternatives. Error-transition analyses in contrastive decoding further motivate reporting corrected and damaged answers separately (Lin et al., 2026). We apply both interventions to the same trained language-LoRA checkpoint to distinguish inference changes from additional training.

Training and evaluation design. SpeLangy isolates data-processing choices and contamination, while SALAD compares additional-stage learning with continued training under a controlled budget (Udandarao et al., 2026; Cuervo et al., 2026). These precedents motivate shared source groups, answer-only supervision, and separate exposure-matched and work-matched comparisons. Thinking while Listening evaluates explicit text/speech reasoning and response latency (Shih et al., 2026). AuralLoop instead produces a short answer after complete-prompt prefill; its hidden states and time to first token (TTFT) have different meanings from generated reasoning traces and spoken-response latency.

Table 5: Related mechanisms and the question addressed by each comparison.

Route	Representative work	Controlled question
Language adaptation	LoRA; Omni-R1	Does language-side adaptation suffice?
Static acoustic access	PAL; DeepStack	Are existing intermediate features sufficient?
Question conditioning	PCLM	Does one-pass question-conditioned mixing suffice?
Direct acoustic adaptation	Audio-tail LoRA	Can ordinary encoder adaptation replace re-entry?
Inference intervention	AAD; IAAN	Can inference changes recover similar gains?
Active re-listening	Echo	How does latent prefill re-entry differ from generated segment access?

B NOTATION AND TENSOR INTERFACES

Acoustic positions before pooling, projected positions after pooling, and language-token positions are distinct index spaces. Table 6 specifies the interfaces between them. Calculations with an omitted clip index apply independently to each clip.

Table 6: Notation and shapes for the 7B configuration.

Symbol	Shape / type	Definition
$x = (x_1, \ldots, x_m)$	m waveforms	Ordered input clips.
q, y	token sequences	Complete observed prompt and answer including its end token.
S_ψ	frozen map	Audio frontend and encoder layers 1–24.
T_ϕ	frozen map	Encoder layers 25–32, native reconstruction, pooling, post-normalization, and projection.
μ_i	metadata	Valid lengths/masks, chunk boundaries, position bookkeeping, and pooling alignment.
C_i^k	$T_{c,i} \times d_a$	Pre-pooling acoustic state; $d_a = 1280$.
E_i^k	$T_{a,i} \times d_\ell$	Projected audio embeddings; $d_\ell = 3584$.
$H^k, \bar{H}^k$	$n \times d_\ell$	Full token stream after/before injection, with n packed positions.
$\mathcal{A}_i, a_{i,j}$	indices	Ordered audio slots and the slot for projected row j.
p	token index	Final complete-prompt position, preceding all answer tokens.
$T_y, \mathcal{D}$	integer, dataset	Answer length including EOS, and the adaptation set.
$p_\Theta, \hat{y}$	distribution, sequence	Conditional answer distribution and greedy predicted answer.
L	28	Number of language layers.
b_1, b_2	$8, 18$	Language-layer boundaries.
r^k	d_ℓ	Column-vector state at p after language stage k.
$D_{a:b}^{\omega,\eta}$	causal map	Language layers a through b with base weights and LoRA.
d_b	256	Shared-cell bottleneck width.
u_t, z^k, m_t^k	d_b	Acoustic projection, language projection, and their product.
g_t^k, α_k	scalars	Temporal gate and step-specific anchor-update scale.
γ_k	d_ℓ	Step-specific injection gate.
$s_{i,j}$	positive scalar	Detached RMS reference from $e_{i,j}^0$.
$\Delta E_i^k, \delta e_{i,j}^k$	matrix, vector	Embedding difference and its transposed row j.
J_k	scatter-add	Boundary-k update to existing audio-token rows.
θ	parameters	Shared projections, affine LayerNorms, and temporal gate.
η, Θ	parameters	Language LoRA; all trainable parameters.
ψ, ϕ, ω	parameters	Frozen stem, tail/projector, and language backbone.
$\epsilon, \epsilon_{\text{opt}}$	$10^{-6}, 10^{-8}$	RMS regularizer and AdamW numerical constant.
ϵ_{LN}	10^{-5}	Numerical constant in both new affine LayerNorms.
$\sigma, \odot$, sg	operators	Sigmoid, elementwise product, and stop-gradient.

Conventions and broadcasting. Matrices store positions as rows. Lower-case feature vectors are their transposes. Thus $c^k_{i,t} = (C^k_{i,t,:})^\top$ and $e^k_{i,j} = (E^k_{i,j,:})^\top$. The injection in Equation (5) transposes its column-vector residual before updating a row of H. The scale $s_{i,j}$ broadcasts across language channels, g^k_t across acoustic channels, and z^k across valid acoustic positions. The native pooling map, rather than broadcasting, relates $T_{c,i}$ to $T_{a,i}$. The projection shapes are $W_a\colon\ d_b \times d_a$, $W_r\colon\ d_b \times d_\ell$, and $W_o\colon\ d_a \times d_b$, with $w_g \in \mathbb{R}^{d_b}$. Both affine LayerNorms normalize channels within each position. We use $\|\cdot\|_2$ for the Euclidean vector norm, $\|\cdot\|_\infty$ for the maximum absolute entry, and $\|\cdot\|_F$ for the Frobenius matrix norm.

Physical token indices. Decoder tensor indices p and $a_{i,j}$ are zero-based, ranging from 0 to $n-1$. For an unpadded example with native packed prompt length L_q, the final prompt index is $p = L_q - 1$. If packing or left padding adds o preceding positions, the stored index becomes $p = o + L_q - 1$. The same offset shifts the owning example's audio-slot map. Native position IDs are maintained separately from these physical tensor indices.

Length and gradient interfaces. Every re-entry preserves the clip's valid lengths and original chunk metadata. Scattering uses the original post-pooling audio-slot map, including partially filled chunks. Frozen tail weights retain derivatives with respect to their inputs. Parameter freezing does not detach the computation graph. The fixed anchor and initial embeddings can be cached outside autograd. The reference RMS is detached explicitly, while r^k remains differentiable in the default model.

The decoder notation suppresses the native causal and padding masks, positional indices, and sample-isolation masks used for packed examples. These interfaces remain unchanged at every language stage. Audio masks remain in μ_i and preserve clip boundaries.

C Properties of the computation graph

The results assume deterministic evaluation in exact arithmetic with finite numerical activations. Parameters, prompt layout, positions, masks, chunk metadata, and sample boundaries are fixed. Stochastic executions are compared conditional on identical call-specific random states.

Proposition 1 (Zero-injection reduction)**.** *If $\gamma_1 = \gamma_2 = 0$, AuralLoop has the same output distribution as the original encoder–language pipeline with the same language-LoRA parameters η. If the LoRA updates also vanish, it recovers the released backbone.*

Proof. Equation (5) becomes $H^k = \bar{H}^k$ at both boundaries. The language stages therefore compose into the ordinary decoder applied to the unchanged initial embeddings E^0 and prompt. The unused re-entry branches do not affect token states. Setting the remaining LoRA updates to zero recovers the original weights. □

Evaluation-mode reduction tests check equality of the output distributions, rather than agreement in benchmark accuracy.

Proposition 2 (Answer-prefix invariance)**.** *For a fixed observed input (x, q) and unchanged prompt layout, replacing teacher-forced answer-token content leaves $r^1, r^2, C^1, C^2, E^1, E^2$ unchanged, provided p precedes every answer token and attention is causal within each language stage.*

Proof. Causal attention prevents prompt positions in the first language stage from reading any answer token, so r^1 depends only on (x, q). The first re-entry and its injection are consequently unchanged. The same causal argument applies to prompt positions in the next stage, making r^2 and the second re-entry unchanged. Answer positions retain their ordinary dependence on the permitted answer prefix. □

Complete-question information can reach earlier audio positions through the feedback path. This makes the prompt a conditional input encoding while preserving autoregressive generation of y_u given $(x, q, y_{<u})$.

Proposition 3 (Conditional information ceiling)**.** *Let X, Q, and M be random variables for the waveform collection, observed prompt, and retained waveform-dependent metadata. Let C^0 collect all cached clip states and define $A^0 = (C^0, M)$. At fixed parameters, each revised encoder-output collection Z is a deterministic function $Z = f(A^0, Q)$ and*

$$I(X; Z \mid Q) \le I(X; A^0 \mid Q). \tag{7}$$

Here $I(\cdot;\cdot \mid \cdot)$ denotes conditional mutual information.

Proof. The native embeddings are deterministic functions of A^0, including the metadata used by the tail. Each language state and subsequent re-entry is therefore a function of (A^0, Q). Conditional data processing yields the result. □

Including M is essential because waveform length and chunk layout can depend on X. Conditioning additionally on M gives $I(X; Z \mid Q, M) \le I(X; C^0 \mid Q, M)$. This structural inequality does not quantify information preservation, and mutual information can be infinite for deterministic continuous variables. For randomized inference, the argument conditions additionally on auxiliary randomness independent of the input.

C.1 Incremental revisions and their magnitude

Encoder-space differences do not telescope through intervening language computation. For one clip, let H be a compatible language state, D an intervening nonlinear stage, and J a fixed linear scatter-and-scale map. Then

$$D\big(H + J(E^1 - E^0)\big) + J(E^2 - E^1) \neq D(H) + J(E^2 - E^0) \tag{8}$$

in general. Distinct step gates can further prevent cancellation. The second residual specifies the change relative to the preceding encoder output. It does not remove the first update's downstream effect. The non-incremental $E^2 - E^0$ variant therefore tests an alternative update rule.

The injection at each valid audio row obeys

$$\|H^k_{a_{i,j},:} - \bar{H}^k_{a_{i,j},:}\|_2 \le \|\gamma_k\|_\infty \frac{\|e^k_{i,j} - e^{k-1}_{i,j}\|_2}{s_{i,j}}. \tag{9}$$

This follows from the standard elementwise-product norm bound. Because $s_{i,j} \ge \sqrt{\epsilon} > 0$, the denominator is nonzero. The fixed reference avoids dividing an update by a scale estimated from that update itself. For fixed reference scales and channel gates, the injected update is linear in the embedding difference. Only g^k_t is sigmoid-bounded. The learned scales α_k and channel gates γ_k are unconstrained. Because the embedding differences and learned gates are not bounded by this construction, the inequality alone does not establish global stability.

C.2 Condition reuse and finite-depth feedback

If $r^2 = r^1$ and $\alpha_2 = \alpha_1$, the shared deterministic cell and tail yield $C^2 = C^1$ and $E^2 = E^1$. The second residual then vanishes exactly. Different step-specific α_k need not produce this collapse. This is why the reuse intervention is interpreted alongside explicit second-update suppression and a norm-matched donor intervention (Appendix N.2).

Two calls form a finite, depth-unrolled graph without convergence or equilibrium constraints. Zero injection embeds the same-LoRA backbone in the model family, and the second tap depends on the first revision. These structural properties do not guarantee better optimization or monotone improvement. Paired-error and direct-adaptation comparisons assess their empirical value.

D Forward algorithm and cache semantics

Algorithm 1 specifies prompt prefill. Clips remain separate inside the acoustic encoder. The language model forms cross-clip relations after native packing.

The copy denotes a separate scatter-add result, preserving $\bar{H}^k$ for autograd and any shadow branch. The algorithm returns the distribution for the first answer token from the final head at the fixed prompt-end position p.

Algorithm 1 Two-step AuralLoop prompt prefill

Require: Clips x_i and metadata μ_i for $i = 1, \ldots, m$, prompt q, boundaries $(b_0, b_1, b_2) = (0, 8, 18)$, and $L = 28$ language layers

for each clip i **do**
  $C_i^0 \leftarrow S_\psi(x_i; \mu_i)$; $E_i^0 \leftarrow T_\phi(C_i^0; \mu_i)$
  $s_{i,j} \leftarrow \mathrm{sg}(\sqrt{\|e_{i,j}^0\|_2^2/d_\ell + \epsilon})$ for valid rows j
end for
$(H^0, \{\mathcal{A}_i\}, p) \leftarrow \mathrm{NativePack}(\{E_i^0\}, q)$
for $k = 1, 2$ **do**
  $\bar{H}^k \leftarrow D^{\omega,\eta}_{b_{k-1}+1:b_k}(H^{k-1})$; retain this stage's keys and values
  $r^k \leftarrow (\bar{H}^k_{p,:})^\top$
  for each clip i **do**
    $C_i^k \leftarrow F_\theta(C_i^0, r^k; \alpha_k)$ ▷ Reset to the original anchor
    $E_i^k \leftarrow T_\phi(C_i^k; \mu_i)$; $\Delta E_i^k \leftarrow E_i^k - E_i^{k-1}$
    $\delta e_{i,j}^k \leftarrow (\Delta E_{i,j,:}^k)^\top$ for valid rows j
  end for
  $H^k \leftarrow \mathrm{copy}(\bar{H}^k)$
  For every clip i and valid row j, add $(\gamma_k \odot \delta e_{i,j}^k / s_{i,j})^\top$ at $a_{i,j}$
end for
Run layers $b_2 + 1 : L$, final normalization, and output head; retain the remaining KV states
return First-answer distribution from output logits at p, and all layer-wise prompt KV states

Layer-wise caches. Layers 1–8 form their key–value (KV) cache before the first injection. Layers 9–18 process the first revised input, and layers 19–28 process the second. A later injection does not overwrite earlier-layer caches. Each generated answer token traverses the full language stack and attends to the corresponding cache at each layer. Recomputed-prefix and cached decoding use the same fixed prompt tap p throughout generation.

Teacher forcing and packed batches. Training may process prompt and answer tokens together under causal attention. Each example stores the prompt-end index before appending the answer, so the tap never becomes the last answer position. Packed batches retain per-example taps and attention masks, per-clip audio-slot maps, and valid lengths. Acoustic chunks cannot attend across clip boundaries. Next-token label shifting aligns each target token with the preceding position's logits, beginning with y_1 at p. The global loss uses actual example counts for partial batches, as specified in Appendix H.

New questions. The question-independent C^0 and E^0 can be reused under unchanged acoustic preprocessing and metadata. Each new prompt is packed with its own audio-slot map and prompt-end index. Revised embeddings and upper-layer KV states are question-dependent and must be recomputed. The evaluated serving regime rebuilds all prompt KV states, performs both re-entries during prefill, and then uses ordinary cached decoding.

E CELL IMPLEMENTATION AND GRADIENT FLOW

The shared cell operates before pooling (Figure 5). Re-entry resumes from the cached layer state without repeating the frontend or positional embedding. The frozen tail retains native output reconstruction, pooling, post-normalization (`ln_post`), and projection.

Initialization. Bias-free cell projections use Xavier-uniform initialization. Both affine LayerNorms use $\epsilon_{\mathrm{LN}} = 10^{-5}$, unit scales, and zero biases. Native norms are unchanged. We initialize $w_g \sim \mathcal{N}(0, 10^{-4} I_{d_b})$, $\alpha_1 = \alpha_2 = 0.1$, and $\gamma_1 = \gamma_2 = 10^{-3}$ in every channel, where I_{d_b} is the identity matrix. Language LoRA uses a Kaiming-uniform input factor and zero output factor, giving zero initial effective weight update.

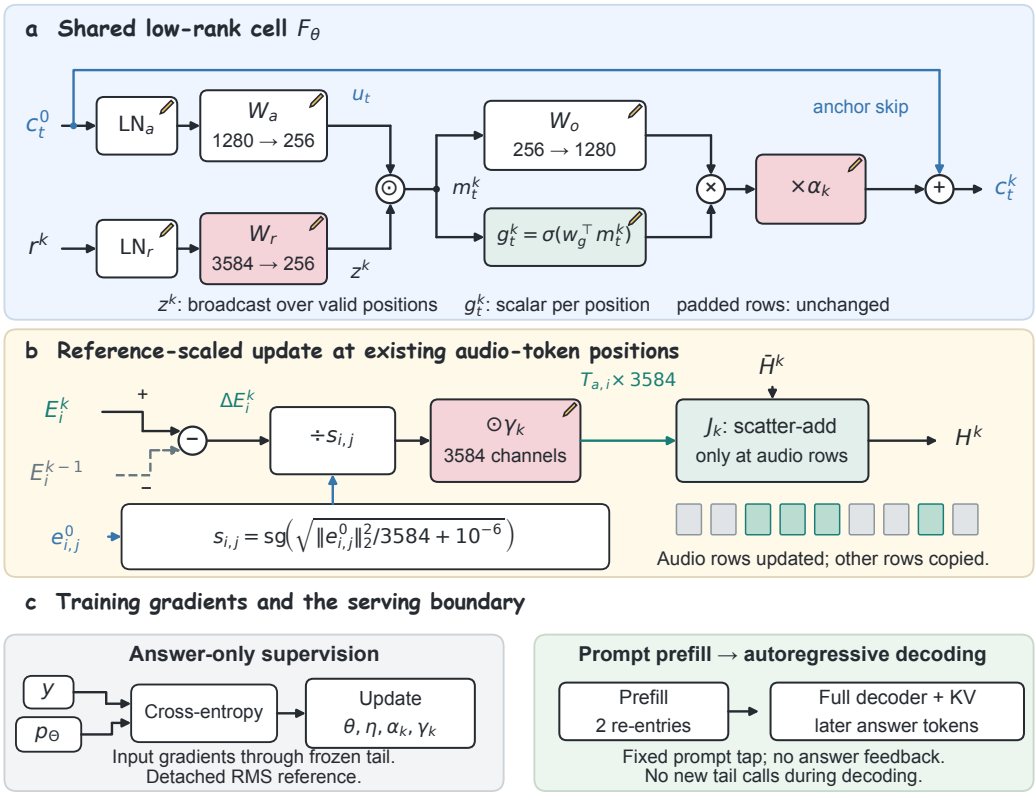

Figure 5: **Cell, injection, and execution interfaces.** (a) The shared cell combines acoustic and broadcast language projections, gates the update, and adds it to the original anchor. Both LayerNorms are trainable. (b) Differences use the dashed previous-embedding input, then reference scaling and audio-slot scatter. The RMS reference is stop-gradient, and C^0, E^0 are cached without autograd. The second difference retains gradients through E^1. (c) Answer-only loss trains the added parameters. Both re-entries occur during prefill, followed by full-decoder generation with layer-wise KV caches.

```
# C0: [B, Tc, da]; r: [B, dl]; valid: [B, Tc]
u = Wa(LN_audio(C0))                    # [B, Tc, db]
z = Wr(LN_language(r))[:, None, :]      # [B, 1, db]
m = u * z
position_gate = sigmoid(m @ wg)         # [B, Tc]
update = Wo(m) * position_gate[..., None]
Ck = preserve_padding(C0 + alpha[step_idx] * update, C0, valid)
Ek = frozen_tail_with_input_grad(Ck, chunk_metadata)
delta = Ek.float() - E_previous.float()
H = scatter_add_audio(H_before, gamma[step_idx] * delta / scale0,
                      audio_position_map)
```

Here k is the mathematical step and `step_idx`$= k-1$ its zero-based array index. The symbol `B` is batch size, `H_before` is $\bar{H}^k$, and `scale0` broadcasts $s_{i,j}$ over channels. Packed chunks gather the owning example's condition. Native packing and position handling are omitted.

Autograd and numerical scales. Revised tails retain input derivatives, and r^1, r^2 connect to language LoRA. Detaching these states defines the stop-gradient ablation. Only LoRA dropout is active, with its RNG states preserved during activation checkpointing.

Anchor changes, embedding differences, and gated injections have distinct magnitudes. Nonzero gates allow a structural derivative path in exact arithmetic but do not guarantee a nonzero gradient at every input or parameter value. BF16 write-back can also absorb a small residual. These implementation effects are distinct from the local injection bound in Equation (9).

F EXPERIMENTAL WORKFLOW

Figure 6 connects the shared adaptation protocol, comparison families, and evaluation outputs. Separately trained configurations and interventions on a fixed checkpoint address different experimental questions.

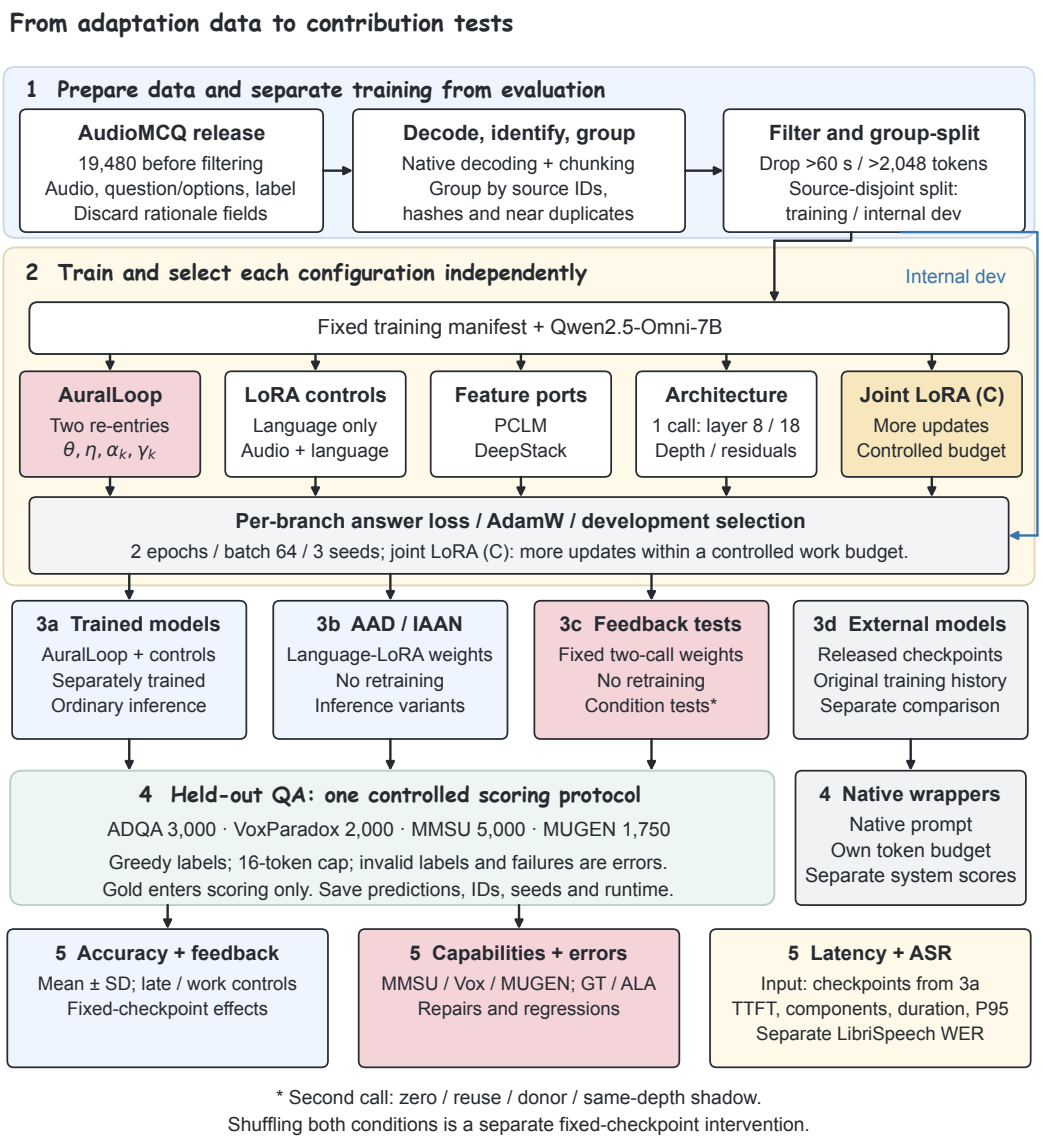

Figure 6: **From adaptation data to controlled evaluation.** Data preparation and grouped development selection precede held-out evaluation. The training branches compare adaptation scope, encoder access, call placement, and training work. Fixed-checkpoint branches change feedback conditions at inference. Benchmark, capability, paired-error, and runtime analyses characterize complementary aspects of the resulting models. Latency and ASR use checkpoints from branch 3a under separate workloads. Gold answers supervise training and enter only scoring at evaluation. They do not condition re-entry. GT denotes ground-truth-label accuracy. SD is the sample standard deviation across training seeds. P95 is the 95th percentile of paired TTFT ratios.

G DATA, SOURCE GROUPING, AND PREPROCESSING

Adaptation resource. AudioMCQ-StrongAC-GeminiCoT is a released 19,480-record subset (Harland, 2026b; He et al., 2026a). We retain audio, questions, choices, and final labels, discarding the rationale field. The source composition in Table 7 describes the released resource before filtering. Answer-only adaptation does not remove the teacher-model involvement in the resource's original construction.

Table 7: Source composition reported by the adaptation dataset card.

Source	Examples
SpeechCraft	9,603
AudioCaps	4,705
CompA-R	2,901
TACOS	1,128
LP-MusicCaps-MTT	807
Clotho	194
MusicCaps	142
Total	19,480

Partitioning and sample identity. Source identity, exact canonical-waveform hashes, and verified duplicate links define connected audio-source groups. Whole groups are assigned to training or development, balancing source and question type, yielding 17,872 training and 1,440 development records. Stable SHA-256 keys combine split seed 20260803 with sorted member IDs. Source components remain intact across the split. Public evaluation audio and its verified source components are excluded from adaptation.

Filtering and exact counts. Table 8 gives the sequential first-failure counts. Each exclusion is counted only among records surviving the preceding stages, so the categories are mutually exclusive. Their 168 exclusions leave 19,312 examples. Distinct questions about the same recording can remain as distinct examples within one source group.

Table 8: Sequential filtering of the 19,480-record adaptation resource.

Stage	Removed	Remaining
Released resource	—	19,480
Empty or undecodable audio	8	19,472
Invalid answer schema	5	19,467
Audio longer than 60 seconds	87	19,380
Complete native template above 2,048 tokens	29	19,351
Verified source link to public evaluation audio	39	19,312
Grouped training split	—	17,872
Grouped development split	—	1,440

Near-duplicate review. Exact matches use source IDs and hashes of canonical mono 16-kHz waveforms. For recordings whose durations differ by at most the larger of 0.1 seconds and 1% of the shorter duration, we propose near-duplicate candidates when the peak absolute normalized correlation reaches 0.995 within a ± 0.5-second alignment window. Potential crop pairs require a source-ID, timestamp, or retrieval link. The aligned overlap must reach the same correlation threshold over at least $\min(2\,\mathrm{s}, 0.9T_{\mathrm{short}})$, where T_{short} is the shorter recording duration, with a unique, non-silent alignment. A candidate becomes a source-group edge only after source/crop metadata and listening confirm the same recording or a transformed copy. Similar wording, speaker identity, or musical content alone does not create an edge. Connected components propagate the resulting train/development separation and evaluation exclusions. Candidate retrieval and review use recording identity and signal content, without test labels or model correctness. This procedure addresses identified source overlap, not unknown overlap in backbone pretraining.

Canonical decoding. Audio channels are averaged to mono and resampled to the native processor's 16-kHz rate. The processor supplies feature extraction, chunking, and padding. We apply no per-clip loudness normalization, denoising, speed modification, or transcript assistance. Sample identity includes the raw-file hash, canonical-waveform hash, source sample rate, channel count, decoded length, source ID, and QA-to-clip mapping. Empty or undecodable recordings and invalid answer schemas are excluded before partitioning.

Length and coverage. The training track excludes clips above 60 seconds and examples whose complete native template exceeds 2,048 packed tokens. Questions and audio are not truncated to satisfy these limits. Evaluation retains complete clips and the official candidate order. Unsupported inputs, invalid outputs, and runtime failures remain in the full evaluation denominator, with their failure types reported separately.

Benchmark identity and dependence. ADQA contains source material derived from other benchmarks, including MMSU (Harland, 2026a). Scores are therefore reported by benchmark without treating all four test sets as independent replications. Our MMSU is the 5,000-question, 47-task perception/reasoning suite (Wang et al., 2026). SALAD uses a different VoiceBench-related setup with synthesized audio and alternative prompting/scoring (Cuervo et al., 2026); those published values are not interchangeable with this suite.

Resource identities and data-scale subsets. The adaptation resource is `Harland/AudioMCQ-StrongAC-GeminiCoT`, ADQA uses `Harland/ADQA-Bench`, and MMSU uses `ddwang2000/MMSU`. The ADQA evaluation contains 3,000 questions. VoxParadox uses its 2,000-question, ten-category release, and MUGEN uses 1,750 questions with five candidate clips and a separately identified reference. The data-scale experiment uses nested whole-group subsets of 1,984, 4,992, 9,984, and 17,872 training examples, with the same 1,440-example development set throughout. Figure 7 uses these exact subset sizes.

Data use. Source-audio and model licenses govern redistribution independently of public availability. Reproducibility materials can identify restricted recordings through metadata and checksums without redistributing them. Demographic and affective benchmark categories are treated as task annotations, rather than verified personal attributes for consequential decisions.

H Training and Model Selection

All controlled adaptations use the same source groups, answer-only objective, option-permutation rule, and development criterion. The principal comparison matches data exposure. The additional joint-LoRA control receives a separately specified training-work allowance.

Loss and update normalization. We first average answer-token losses within each example, including the native end token, then average over valid examples across ranks and accumulated microsteps. After gradient synchronization, we clip trainable gradients, update the optimizer, and advance the scheduler. Partial final batches are normalized by their actual global count of valid examples. Masked dummy entries can equalize distributed shapes but receive zero weight.

Update counts and learning-rate schedule. Each epoch contains 279 full 64-example updates and one 16-example update, because $17{,}872 = 279 \times 64 + 16$. Two epochs therefore complete 560 updates and expose 35,744 examples. For update $u \in \{1, \dots, U\}$, the learning rate for each parameter group is

$$\mathrm{lr}(u) = \mathrm{lr}_{\mathrm{base}}\, \lambda_{\mathrm{LR}} \begin{cases} u/W, & 1 \le u \le W, \\ \left[1 + \cos\!\big(\pi(u-W)/(U-W)\big)\right]/2, & W < u \le U, \end{cases} \qquad W = \lceil 0.05U \rceil. \tag{10}$$

The group base rates are in Table 9, and λ_{LR} is the search multiplier. The equal-exposure validation updates are 50, 100, 150, 200, 250, 280, 300, 350, 400, 450, 500, 550, and 560. The smaller data-scale subsets use 62, 156, and 312 updates, with 4, 8, and 16 warm-up updates, respectively.

Table 9: Default AuralLoop training configuration. Baseline-specific adapters are detailed in Appendix I.

Setting	Value
Backbone	Qwen2.5-Omni-7B Thinker
Frozen modules	Base encoder, projector, Thinker, embeddings, and output head
Trainable modules	Shared cell, step-specific α_k/γ_k, and Thinker q/o LoRA
Language LoRA	Rank 4; scaling numerator 8; dropout 0.05
Cell and boundaries	$d_b = 256$; anchor after audio layer 24; language boundaries 8/18
Initialization	Xavier-uniform cell maps; affine LN scale 1/bias 0, epsilon 10^{-5}; $w_g \sim \mathcal{N}(0, 10^{-4} I_{d_b})$; $\alpha_k = 0.1$, $\gamma_k = 10^{-3}$
AdamW	Betas 0.9/0.95; epsilon 10^{-8}; weight decay 0.01
Learning rates	Cell/new interfaces/gates: 2×10^{-4}; audio/language LoRA: 8×10^{-5}
Decay exclusions	Normalization parameters, scalar/channel gates, and biases
Batch and duration	Global batch 64; two epochs, 560 updates; training audio at most 60 s
Context and bucketing	At most 2,048 packed tokens; sortish windows of 256 examples
7B execution	Eight GPUs; microbatch 1 per rank; accumulation 8
3B engineering configuration	Two GPUs; microbatch 1 per rank; accumulation 32
Schedule	28 warm-up updates, then cosine decay to zero for the 560-update track
Numerics	BF16; FP32 loss/reductions; trainable-gradient clipping at 1.0
Memory	FSDP full sharding; activation checkpointing; no CPU offload or training KV cache
Dropout	Frozen modules disabled; LoRA dropout active during training
Validation	Every 50 updates and epoch end; highest accuracy, earlier-checkpoint tie break
Seeds	Training: 3407, 7189, 12011; grouped split: 20260803

Search and selection. Each trainable comparator uses the same learning-rate multiplier grid $\{0.5, 1, 2\}$ on grouped internal development data. Ties prefer the smaller multiplier and then the earlier checkpoint. The selected multiplier is 1 for all reported trainable configurations. Development selection yields update 560 for each full-data equal-exposure run, including the static ports and the independently trained depth, call-count, and structural variants. The selected Joint LoRA (C) updates are 712, 717, and 710 for seeds 3407, 7189, and 12011. The smaller data-scale runs select their respective final checkpoints. The architecture search covers the declared anchor depths and call counts. Public test results do not select checkpoints, boundaries, or learning rates. AAD and IAAN inherit each seed's selected language-LoRA checkpoint. Same-checkpoint interventions inherit the selected two-call checkpoint without further training or selection.

Adapter scaling. The language rank-4 adapter uses scaling numerator 8, giving an effective multiplier of $8/4 = 2$. Rank-8 language-only LoRA uses numerator 16, and rank-20 audio LoRA uses numerator 40. All projection biases remain frozen. Trainable-parameter matching allows a 2% tolerance and does not imply identical execution cost.

Execution configuration. The specified 7B configuration uses eight 24-GB RTX 4090 GPUs with full parameter sharding and sharding-aware initialization. The software stack is PyTorch 2.7.1, Transformers 4.53.3, and PEFT 0.16.0, using the eager attention backend. The model is `Qwen/Qwen2.5-Omni-7B` with its native processor, tokenizer, and chat template. Activation checkpointing preserves LoRA RNG states. Deterministic interface comparisons disable dropout, while training keeps the declared LoRA dropout active. Native chunk masks and post-pooling normalization remain unchanged. Re-entry resumes from the cached encoder state without adding the front-end positional embedding again. The analytic parameter and payload counts below describe the specified interfaces. Runtime memory additionally depends on packing and sharding.

H.1 Training-work-controlled joint adaptation

Joint LoRA (C) starts from the same pretrained checkpoint and retains audio rank 20 and language rank 4. Before training, its total update counts are fixed at 712, 717, and 710 for seeds 3407, 7189, and 12011, respectively, with 36 warm-up updates and cosine decay ending at the corresponding horizon. It receives additional updates from the same training stream. For method M and training seed s, let $w_{M,s}(u)$ be the measured wall-clock seconds for optimizer update u across the same

synchronized eight-GPU configuration. For $U_{M,s}$ completed updates, accumulated training work is

$$B_{M,s} = \sum_{u=1}^{U_{M,s}} w_{M,s}(u), \qquad M \in \{2, C\}. \tag{11}$$

Here $M = 2$ denotes two-call AuralLoop and $M = C$ denotes Joint LoRA (C). An update's timed interval starts with input transfer for its first microstep and ends after gradient synchronization, global gradient clipping, optimizer and scheduler updates, and synchronization of all devices. It includes forward computation, encoder re-entry where applicable, backward computation, activation-checkpoint recomputation, and FSDP communication. The CPU feature tensors for all microsteps are prepared before this interval. Waveform decoding, CPU feature extraction, data-read waiting before the interval, validation, checkpoint saving, and one-time loading or compilation are excluded. Neither method caches acoustic prefixes across examples or epochs. This metric measures the synchronized training intervals, rather than the entire data-processing and training project.

Table 10: Measured cumulative training work and completed optimizer updates. The remaining allowance is $B_{2,s} - B_{C,s}$, in seconds.

Seed	$U_{2,s}$	$B_{2,s}$ (s)	$U_{C,s}$	$B_{C,s}$ (s)	Remaining (s)
3407	560	5491.8244	712	5486.4496	5.3748
7189	560	5681.7756	717	5675.3424	6.4332
12011	560	5597.9736	710	5591.4852	6.4884

The completed runs in Table 10 remain below their paired allowances by less than 0.12%. Joint LoRA (C) sees 45,472, 45,792, and 45,344 examples across the three seeds, respectively. Its additional updates end within the third epoch, after full batches. The control thus tests whether additional joint-adapter optimization on the same source data explains the benefit of re-entry.

Checkpoint selection. The extended schedule uses the same development criterion, learning-rate search, and tie rule as the equal-exposure track. Equation 10 uses each run's predetermined horizon as U throughout training. Both tracks have 13 validation opportunities. Joint LoRA (C) aligns these opportunities to normalized cumulative training-work progress, taking the first completed update reaching each target and ending at the scheduled final update. Its validation updates are listed in Table 11. The selected checkpoints are the final scheduled checkpoints for the reported runs. The measured work comparison in Table 10 applies to each completed training run. The three learning-rate candidates have the same selection opportunities, and their search cost is not included in a single run's budget. Training work and deployment latency remain separate quantities: the work-controlled joint adapter uses its ordinary single-pass architecture at inference.

Table 11: Joint LoRA (C) validation updates under the training-work allowance.

Seed	Validation updates
3407	64, 128, 191, 255, 319, 357, 382, 446, 510, 574, 638, 701, 712
7189	65, 129, 193, 257, 322, 359, 385, 449, 513, 578, 642, 706, 717
12011	64, 127, 191, 254, 319, 356, 381, 445, 508, 572, 636, 699, 710

Deployment profiling. Linear audio and language LoRA weights are merged before profiling at the same deployment precision (Hu et al., 2022). Input-dependent encoder re-entry remains active at inference. The systems configuration is detailed in Appendix P. Adapter parameter count therefore does not by itself determine deployed latency.

H.2 Parameter counts and cache payload

A bias-free rank-r LoRA update to a $d_{\text{in}} \to d_{\text{out}}$ map has $r(d_{\text{in}} + d_{\text{out}})$ trainable parameters. Table 12 applies this rule to the specified interfaces. The language-only and joint controls have slightly larger trainable budgets than AuralLoop. Adding one active re-entry introduces only a scalar α_k and $d_\ell = 3584$ injection channels. One-, two-, and three-call totals are therefore 3,192,065, 3,195,650, and 3,199,235. The cell, frozen tail, and their weight storage are shared across invocations.

Table 12: Analytic trainable-parameter counts.

Component	Parameters
$W_a : 1280 \to 256$	327,680
$W_r : 3584 \to 256$	917,504
$W_o : 256 \to 1280$	327,680
Affine LayerNorms, $2(1280 + 3584)$	9,728
Temporal gate vector	256
Two anchor gates	2
Two channel-injection gates, 2×3584	7,168
Shared cell and gates	1,590,018
Language q/o LoRA, $28 \times 2 \times 4 \times (3584 + 3584)$	1,605,632
AuralLoop total	**3,195,650**
Language-only LoRA, rank 8	3,211,264
Audio-tail q/k/v/o LoRA, $8 \times 4 \times 20 \times (1280 + 1280)$	1,638,400
Joint audio rank 20 + language rank 4	3,244,032

Anchor payload. Using an idealized rate of 50 pre-pooling positions/s, a 60-second BF16 anchor requires

$$60 \times 50 \times 1280 \times 2 = 7{,}680{,}000 \text{ bytes} \approx 7.32 \text{ MiB}. \tag{12}$$

At approximately 25 post-pooling positions/s, the initial embeddings require $60 \times 25 \times 3584 \times 2 = 10{,}752{,}000$ bytes. Their combined ideal payload is approximately 17.58 MiB. Actual valid lengths and chunk padding determine the tensor allocation. Metadata, activations, allocator overhead, and distributed states contribute additional memory.

I Baseline implementations and structural controls

The controlled track distinguishes adaptation scope, ports of published mechanisms, and inference interventions. Released complete checkpoints are evaluated separately because their pretraining and post-training differ.

Table 13: Adaptation scope on the shared Qwen2.5-Omni backbone.

Method	Acoustic interface	Language/decision interface
LoRA (LLM)	Frozen native path	All Thinker q/o, rank 8
LoRA (audio+LLM)	Tail 25–32 q/k/v/o, rank 20	All Thinker q/o, rank 4
DeepStack†	Static layers 8/16/24/32	Static injections and rank-4 LoRA
PCLM†	Question-weighted layers 5/15/25/30/32	Mixer and rank-4 LoRA
AAD on LoRA (LLM)	Audio and reference conditions	Contrastive decoding
IAAN on LoRA (LLM)	Encoder-neuron intervention	Same trained language-LoRA checkpoint
AuralLoop	Shared cell; frozen-tail re-entry	Rank-4 LoRA; incremental audio-slot updates

PCLM port. The port reads encoder layers $\{5, 15, 25, 30, 32\}$. Each branch uses native chunk-wise pooling, normalization, and a frozen projector, plus a rank-64 acoustic-to-language residual map. The frozen prompt encoder is `google/bert_uncased_L-4_H-512_A-8`, with four layers, a hidden size of 512, and eight attention heads. Its final-layer `[CLS]` states supply a 512-dimensional question/choice representation to a trainable $512 \to 32 \to 5$ mixer. The mixer uses GELU between its linear layers and softmax over the final five outputs. The BERT encoder remains in evaluation mode. Its tokenizer receives the question and choices, without audio placeholders, gold labels, or assistant answers, and has a 512-token context limit. The mixer weights depend on the observed prompt, not the answer label. The residual maps, mixer, and rank-4 language LoRA train jointly under the shared answer-only schedule. This retains question-conditioned layer mixing while differing from PCLM's original staged training and optional DPO (Pang et al., 2026).

Long inputs to the auxiliary prompt encoder. Question and choice text is serialized as one string and tokenized by BERT without special tokens or truncation. Let $L_B > 0$ be its number of content tokens. Empty content is an invalid input schema. We partition the sequence into ordered, nonoverlapping windows of at most 510 content tokens. Each receives the native `[CLS]` and `[SEP]` tokens, with positions restarted at zero, keeping every BERT input within 512 positions. For window j, let ℓ_j be its content-token count and $h_j \in \mathbb{R}^{512}$ its final-layer `[CLS]` state. The mixer input is

$$K_B = \left\lceil \frac{L_B}{510} \right\rceil, \qquad \ell_j = \min\{510, L_B - 510(j-1)\}, \qquad v_B = \frac{1}{L_B} \sum_{j=1}^{K_B} \ell_j h_j. \tag{13}$$

Here K_B counts windows, and $j = 1, \ldots, K_B$. For $L_B \leq 510$, the rule reduces to the ordinary single-window state $v_B = h_1$. At 511 content tokens, the two windows contain 510 and one token, respectively. Their weights sum to one without dropping content. All windows share the frozen BERT encoder in evaluation mode, and their evaluations contribute to inference cost. The resulting v_B feeds the same mixer for all acoustic branches, adding no trainable parameters. Windowing affects only the auxiliary BERT branch. The complete question and choices remain in the main language prompt. Length-weighted pooling preserves token coverage but does not provide self-attention across windows.

DeepStack port. The port reads layers $\{8, 16, 24, 32\}$ with native chunk-wise pooling/projectors and rank-80 residual maps. Its four static branches are injected after Thinker layers $\{2, 4, 6, 8\}$ using learned channel gates initialized to 10^{-3}. The branch features are computed before language processing and receive no intermediate language condition. These interfaces and rank-4 language LoRA are trained jointly. The port tests static depth access rather than reproducing MOSS-Audio's full training recipe (Yang et al., 2026a).

Port parameter budgets. Each rank-r acoustic-to-language map has $r(1280 + 3584) = 4864r$ parameters. PCLM therefore uses 1,556,480 map parameters, 16,581 mixer parameters, and 1,605,632 language-LoRA parameters, totaling 3,178,693. DeepStack uses 1,556,480 map parameters, 14,336 channel gates, and the same language LoRA, totaling 3,176,448. Both lie within 1% of AuralLoop's budget. Frozen auxiliary weights do not count as trainable parameters but do contribute storage and inference cost.

Residual-map initialization and optimization. Each added map is $\Delta W = \kappa BA$, where $A \in \mathbb{R}^{r \times 1280}$, $B \in \mathbb{R}^{3584 \times r}$, and the fixed scale is $\kappa = 1$. The maps have no biases. We initialize A with Kaiming-uniform initialization using $a = \sqrt{5}$ and initialize B to zero. The ranks are 64 for PCLM and 80 for DeepStack. The frozen native projector and its post-pooling normalization are unchanged, with no additional trainable dense projector. PCLM's mixer weights use Xavier-uniform initialization and zero biases. DeepStack's four channel gates start at 10^{-3}. Maps, mixers, and gates use the 2×10^{-4} parameter group, while language LoRA uses 8×10^{-5}, with the common schedule and selected multiplier in Appendix H. The custom residual-map scale is independent of the language-LoRA multiplier of two.

AAD under the shared prompt. Our AAD implementation uses each seed's selected language-LoRA checkpoint without additional gradient updates (Hsu et al., 2025). The reference is an all-zero waveform of the same length as the real input. Real and reference branches share the same generated prefix. At each decoding step, let $z_{\text{real}}, z_{\text{blank}} \in \mathbb{R}^V$ denote the real-audio and reference logits over a vocabulary of size V. Greedy selection uses $(1 + \lambda_{\text{AAD}})z_{\text{real}} - \lambda_{\text{AAD}} z_{\text{blank}}$. Development search over $\lambda_{\text{AAD}} \in \{0.5, 1, 2\}$ selects 1. The decoded response uses the controlled track's label parser, without restricting generation to candidate-label tokens. We preserve the common QA prompt rather than adding the dedicated prefix used in the source method. This is a shared-prompt adaptation of AAD, rather than a reproduction of its complete prompting protocol.

IAAN under the shared backbone. Our IAAN implementation also retains each seed's selected language-LoRA checkpoint (Huang et al., 2026). It measures post-nonlinearity encoder-FFN activations on real audio and a same-length Gaussian reference matched to the waveform's mean and variance. For each neuron, the score is the mean post-nonlinearity activation on real audio minus the corresponding mean on the reference, using only valid acoustic frames. The top 100 neurons are

selected globally across all 32×5120 encoder-FFN coordinates, rather than separately within each layer. Development search over gains $\{4, 8, 16\}$ selects $g = 8$, with the neuron budget fixed at 100. Ties in neuron scores follow lexicographic layer and neuron indices.

The reference noise is deterministic for each stable clip identity using seed 9021, and the same reference waveform is reused across training seeds. Each checkpoint computes its own activations and ranking. Multi-clip questions apply identification and amplification separately to each clip before native packing. The output comes from the amplified real-audio pass. Real-audio statistics, reference statistics, and amplified real audio therefore require three encoder passes. Reference processing and intervention computation are included in the inference cost. This fixed-budget search evaluates the stated IAAN port and does not claim an exhaustive search over neuron budgets and gains.

I.1 Controls for update location, timing, and feedback

Projected-token feedback. This variant applies the same multiplicative anchor-reset and incremental-difference design to E^0, bypassing the encoder tail. Replacing d_a with d_ℓ and using bottleneck 144 gives $3 \times 3584 \times 144$ projection parameters. With two affine LayerNorms, a 144-vector temporal gate, and the step gates, the core has 1,569,938 parameters. Rank-4 language LoRA brings the total to 3,175,570. The control tests update location under a similar parameter budget.

Fixed-question conditioning. The frozen native embeddings of question and option tokens are averaged, excluding audio placeholders, role tokens, and padding. This text-only vector conditions both shared-cell calls. The full language stack and both tail calls remain active, with distinct α_k gates. This control supplies the question without incorporating original or revised acoustic evidence into the condition. The step-specific scales are independently trainable. At the stated initialization, $\alpha_1 = \alpha_2$, so the identical fixed condition and deterministic shared cell/tail yield $E^2 = E^1$. The second proposal initially vanishes, and training can subsequently separate the scales.

Early and late single re-entry. The early control injects once after layer 8, and the late control injects once after layer 18. Both retain all 28 language layers and use $E^1 - E^0$ as their only revision. The late condition has processed the original acoustic embeddings through a deeper language stage, but has not observed a preceding revision. It therefore tests whether deeper conditioning alone explains the benefit of two calls. The 3B counterparts use layers 12 and 24. These are independently trained architectures with the same shared-data recipe, not test-time suppression of a two-call checkpoint.

Other architectural variants. The accumulated-anchor variant uses C^1 in both the acoustic projection and the residual reference at the second call:

$$C_i^2 = F_\theta(C_i^1, r^2; \alpha_2), \qquad u_t^2 = W_a \, \mathrm{LN}_a(c_t^1).$$

Thus its second update is $c_t^2 = c_t^1 + \alpha_2 \sigma(w_g^\top m_t^2) W_o m_t^2$, with $m_t^2 = u_t^2 \odot W_r \, \mathrm{LN}_r(r^2)$. The native tail metadata and incremental injection rule are unchanged. The non-incremental variant injects $E^2 - E^0$ at the second boundary. The stop-gradient variant detaches r^k only at the cell input while preserving ordinary language-LoRA gradients. Each variant is independently trained using the shared selection policy. Test-time interventions on fixed weights are defined separately in Appendix N.

External complete systems. MOSS-Audio, AudioMCQ Mixed-to-Strong, and Qwen2-Audio+PCLM+DPO retain their released training histories. Their native wrappers and generation budgets are distinguished from the shared label-only track. Echo is a related audio-revisiting architecture, while ViLaVT is a cross-modal conceptual comparator (Wu et al., 2026a;b). Within-family 3B engineering configurations do not establish transfer to unrelated encoder families.

J Prompts, generation, and scoring

All shared-backbone methods use identical prompt content, native audio insertion, candidate identities, and decoding settings. The templates below document text structure. Angle-bracket placeholders are not additional special tokens. The rendered generation prefix ends at position p before any answer token is appended.

J.1 SINGLE-AUDIO AND MULTI-AUDIO PROMPTS

Single-audio question answering

```
SYSTEM:
Listen to the audio and answer the question.
Return exactly one option label from the listed choices.
USER:
<audio supplied through the native processor>
Question: {question}
{label_1}. {option_1}
...
{label_Kopt}. {option_Kopt}
ASSISTANT:
{native generation prefix}
```

The number of choices K_{opt} comes from the example schema. Training permutations remap choice content and labels jointly. Primary evaluation preserves the original option order. Only the final answer and native end token receive supervised loss.

Multi-audio question answering

```
SYSTEM:
Compare the audio clips and answer the question.
Return exactly one candidate label from the listed choices.
USER:
Reference: <reference clip; only when present>
Candidate A: <audio A>
Candidate B: <audio B>
...
Candidate E: <audio E>
Question: {original question}
ASSISTANT:
{native generation prefix}
```

MUGEN has five answer candidates. A reference retains its own role and is not a sixth option. Candidate permutations jointly update clip placement, identity labels, and the gold mapping. Individual clips remain separate in the encoder.

J.2 DECODING AND EXTERNAL FORMATS

Shared-backbone label QA uses greedy decoding with `do_sample=False`, `num_beams=1`, `max_new_tokens=16`, repetition penalty 1.0, and native end-of-turn/EOS IDs. Sampling-only temperature and top-p parameters are unused. External reasoning checkpoints retain their documented wrappers and generation allowances. Their additional decoding contributes to cost. Native judging and strict generated-label accuracy are reported as distinct protocols.

Speech recognition. The ASR system instruction is "Transcribe the speech faithfully. Return only the transcription." The user supplies audio through the native processor. All compared models decode greedily with a 1,024-token cap, recording any cap hit. Corpus word error rate (WER) sums substitutions, deletions, and insertions across utterances and divides by the total reference-word count. The AL-EN-v1 normalizer applies Unicode NFKC, converts curved apostrophes to straight apostrophes, and uppercases text. It preserves word-internal apostrophes, replaces other punctuation with spaces, collapses whitespace, and splits on spaces. Numbers and abbreviations are not expanded. The same normalization is applied to references and predictions.

Recorded interfaces. The evaluation record retains the rendered prompt, processor/template revisions, option order, valid-label set, prompt-end index, answer span, decoding configuration, and unmodified output. The prompt end is measured immediately after adding the native generation

prefix. Hidden conditions r^k are vectors rather than generated reasoning transcripts, and no analyst interpretation is inserted into model inputs.

J.3 Deterministic answer parsing

The strict generated-label track removes authorized native terminal suffixes and outer whitespace, then accepts exactly one valid case-insensitive label. The label may appear alone, followed by one period, or enclosed in one parenthesis pair. Combined or nested decorations, internal spaces or prose, multiple labels, out-of-range labels, empty outputs, and incomplete outputs are invalid. Terminal markers inside a response are not removed. Invalid labels and runtime failures count as incorrect in the full denominator.

Completion before parsing. A label response must end with a permitted native EOS or end-of-turn token. The evaluator checks the generated token sequence before special-token removal, excluding batch-padding positions. An allowed terminator at the sixteenth generated position counts as completion. Sixteen generated tokens without a terminator count as incomplete, even if the cleaned text matches a valid label. No additional stop strings or custom stopping criteria are used. Timeout, out-of-memory, cancellation, and other runtime failures take precedence over completion and label parsing. Failed and incomplete responses remain in the full accuracy denominator. Only completed responses proceed to the stated grammar and then gold-label comparison.

Table 14: Examples defining the shared generated-label parser. These strings are unit-test inputs.

Response	Result	Rule
` B`	B	Remove outer whitespace.
`(b)`	B	Normalize case and one enclosing parenthesis pair.
`B.`	B	Permit one terminal period.
`(B).`, `(B.)`, `((B))`	Invalid	Combined or nested decorations.
`( B ),B..`	Invalid	Internal whitespace or repeated punctuation.
`B or C`	Invalid	Multiple labels.
`The answer is B`	Invalid	Additional prose.
`E`, four-choice item	Invalid	Label absent from the valid set.
`E`, five-choice item	E	Label present in the valid set.
Cap reached, no EOS/EOT	Incorrect	Incomplete even if the cleaned label is valid.
EOS/EOT at token 16	Parse	Completion is checked before the label grammar.
Empty / timeout / OOM	Incorrect	Retain and record the failure category.

Alternative scoring tracks. Candidate-likelihood scoring is a distinct evaluation: each label sequence is scored using the same observed prompt and a declared length-normalization rule. Appending a candidate does not move the feedback tap. Similarly, verbose-answer extraction and LLM judging are separate protocols rather than substitutions for the strict parser. This distinction matters when comparing source results that use different prompts, resynthesized audio, or native judging (Cuervo et al., 2026; Shih et al., 2026; Yang et al., 2026b).

J.4 Audio-QA examples and diagnostic interpretation

The following cases connect the prompt and scoring protocol to the capability and paired-decision analyses. Case identifiers are local aliases. Audio descriptions and reference labels appear here for interpretation and are not added to model inputs. The native audio placeholders indicate waveform insertion through the processor. Each model produces a final answer after its complete forward pass.

J01: Lexical–acoustic conflict. The speaker says "My voice falls at the end" with a clearly rising final pitch contour. The transcript is shown only to explain the lexical conflict.

```
SYSTEM:
Listen to the audio and answer the question.
Return exactly one option label from the listed choices.
USER:
<native audio: CASE_CLIP_J01>
Question: How does the speaker's pitch change at the end
of the utterance?
A. It falls.
B. It rises.
C. It remains approximately level.
D. There is no voiced ending.
ASSISTANT:
<native generation prefix; no answer token>
```

The reference label is B, while A is the lexical distractor. The model outputs distinguish these labels from a third, incorrect acoustic judgment.

Configuration	Output	Outcome
Language LoRA	A	Incorrect
PCLM port	B.	Correct
Joint audio–language LoRA	C	Incorrect
IAAN on language LoRA	(b)	Correct
Late single re-entry	A	Incorrect
Two re-entries	B	Correct

The late single-call and two-call predictions form a wrong/correct pair between independently trained models. PCLM and IAAN also answer correctly, showing that success on this case is not unique to re-entry. Only the final label and native end token receive training loss. Neither the analyst's pitch description nor the scoring label conditions the re-entry cell.

J02: The same clip with another question. The waveform identity is unchanged from J01. The observed question now asks about spoken content rather than the final contour.

```
SYSTEM:
Listen to the audio and answer the question.
Return exactly one option label from the listed choices.
USER:
<native audio: CASE_CLIP_J01>
Question: Which sentence is spoken in the recording?
A. My voice falls at the end.
B. The bell rings at the start.
C. Two speakers are talking together.
D. The recording contains no speech.
ASSISTANT:
<native generation prefix; no answer token>
```

The reference label is A. Joint LoRA, the late single-call model, and the two-call model return A, a, and A., respectively. All parse to A. For fixed waveform metadata, the cached acoustic anchor can remain unchanged. The new prompt still requires its own packing, prompt-end index, language states, and question-dependent revisions. Reusing an earlier question's upper KV states would not implement the specified computation. Together with J01, this case separates the acoustic question from the lexical one. The two-call model answers both correctly, while the late single-call model succeeds only on the lexical question.

J03: Reference identity and five candidates. Candidate D shares the reference voice but speaks different words. Candidate A repeats the reference's words in a different voice. The remaining candidates use other voices.

```
SYSTEM:
Compare the audio clips and answer the question.
Return exactly one candidate label from the listed choices.
USER:
Reference: <native audio: CASE_REF_J03>
Candidate A: <native audio: CASE_A_J03>
Candidate B: <native audio: CASE_B_J03>
Candidate C: <native audio: CASE_C_J03>
Candidate D: <native audio: CASE_D_J03>
Candidate E: <native audio: CASE_E_J03>
Question: Which candidate is spoken by the same speaker
as the reference?
ASSISTANT:
<native generation prefix; no answer token>
```

The reference label is D. This input has six clips and five answer options. Outputs for language LoRA, PCLM, joint LoRA, late single re-entry, and two re-entries are A, D, D., A, and (D), respectively. PCLM, joint LoRA, and the two-call model all parse to D and identify the matching voice despite different wording. Selecting A instead chooses the same-word, different-speaker distractor. If the candidate order becomes (old D, old A, old E, old B, old C), the correct label becomes A while the reference retains its role. This relabeling explains the identity mapping required by a permutation, without adding another measured prediction. Every clip retains its own anchor and metadata. Native audio-slot maps must be recomputed after reordering.

J04: A counting regression. The recording contains two distinguishable, partly overlapping voices.

```
SYSTEM:
Listen to the audio and answer the question.
Return exactly one option label from the listed choices.
USER:
<native audio: CASE_CLIP_J04>
Question: How many distinct speakers are audible?
A. One.
B. Two.
C. Three.
D. Four.
ASSISTANT:
<native generation prefix; no answer token>
```

The reference label is B. Joint LoRA, IAAN, and the late single-call model return B, whereas the two-call model returns A. This correct/wrong pair illustrates the speaker-counting regression seen in the category analysis. Additional acoustic computation does not ensure that distinct sources are counted correctly. The output alone does not identify which internal operation causes this error.

J05: Fixed-checkpoint intervention. This comparison uses the complete J01 input and reference label B. All rows below share the same two-call checkpoint, its gates, and the receiving sample's first revision. No retraining or new answer head is introduced.

Condition	Output	Outcome
Intact feedback	B	Correct
Zero second injection	A	Incorrect
Same-depth shadow, raw	C	Incorrect
Same-depth shadow, norm-matched	B	Correct

The shadow condition is extracted from the middle language stage applied to $\bar{H}^1$, rather than to H^1. The receiving main stream remains intact. Its alternate proposal subtracts the receiving E^1, not E^0. Norm matching is applied after reference scaling and channel gating, separately for each clip. Here the norm-matched shadow preserves correctness, while the raw shadow and suppressed update fail. The example distinguishes condition replacement from complete path removal and illustrates why

amplitude control matters. Its pattern complements the aggregate intervention results rather than estimating their frequency.

J06: A wrong-to-wrong answer change. The recording contains three short beeps followed by a sustained tone.

```
SYSTEM:
Listen to the audio and answer the question.
Return exactly one option label from the listed choices.
USER:
<native audio: CASE_CLIP_J06>
Question: Which description matches the order of sounds?
A. A sustained tone followed by three short beeps.
B. A sustained tone with no beeps.
C. Two short beeps with no sustained tone.
D. Three short beeps followed by a sustained tone.
ASSISTANT:
<native generation prefix; no answer token>
```

The reference label is D. The late- and two-call answers, B and C, are both incorrect. The pair belongs to wrong/wrong and contributes to neither repair nor regression. Thus correctness-flip rates exclude some changes in answer identity.

J07: Parser boundaries. Table 14 gives the deterministic acceptance tests used by the shared generated-label track. For example, `B.` and `(b)` both map to B, while `(B).` and `The answer is B` are invalid. Label E is valid only when the item's schema includes E. Runtime failure, empty output, and incomplete generation retain their distinct status while counting as incorrect. Thus semantic plausibility does not override the declared response grammar.

J08: A transcription-scoring example. The ASR example uses the same native audio interface with a transcription-specific system instruction.

```
SYSTEM:
Transcribe the speech faithfully. Return only the transcription.
USER:
<native audio: CASE_CLIP_J08>
ASSISTANT:
<native generation prefix; no target text>
```

The reference transcription is "The small bell rang before the train arrived." The model hypothesis is "The bell rang after the train arrived." After AL-EN-v1 normalization, one deletion (SMALL) and one substitution (BEFORE to AFTER) remain. There are no insertions and eight reference words, giving

$$\mathrm{WER} = 100\frac{S+D+I}{N_{\mathrm{ref}}} = 100\frac{1+1+0}{8} = 25\%.$$

Here S, D, I count substitutions, deletions, and insertions, respectively, and N_{ref} is the reference word count. This utterance contributes two errors and eight reference words. Corpus WER sums these quantities across utterances before division, rather than averaging utterance percentages.

J09: A post-scaling norm example. Consider two valid projected rows. Only the first two of their 3584 coordinates are nonzero. The displayed submatrices specify the already scaled and gated intact and replacement residuals:

$$U = \begin{bmatrix} .03 & .04 \\ 0 & 0 \end{bmatrix}, \qquad \widetilde{U} = \begin{bmatrix} 0 & .02 \\ .02 & 0 \end{bmatrix}.$$

Their Frobenius norms are 0.05 and $\sqrt{0.0008}$, respectively. Norm matching multiplies each nonzero replacement coordinate by $0.05/\sqrt{0.0008}$, yielding $0.0353553390593\ldots$. The whole-clip norm becomes 0.05 in exact arithmetic, while its distribution across rows and channels differs from U. If the replacement norm is at most 10^{-8}, the declared zero fallback is used instead of claiming a successful nonzero norm match. The calculation verifies whole-clip amplitude matching while making clear that rowwise or channelwise equality is not enforced.

K PER-SEED ACCURACY COUNTS

Tables 15–17 provide integer correct-answer counts for the listed methods and controls. Extended designs reported only as means are identified separately, while category tables provide three-seed aggregates. For benchmark size N and correct counts c_s, let $s = 1, 2, 3$ enumerate seeds 3407, 7189, and 12011:

$$a_s = 100c_s/N, \qquad \bar{a} = \frac{1}{3}\sum_{s=1}^{3} a_s, \qquad s_a = \sqrt{\frac{1}{2}\sum_{s=1}^{3}(a_s - \bar{a})^2}. \tag{14}$$

Category totals are aggregated before division, and rounding is applied only to final displayed values. Seed standard deviation describes variation between trained checkpoints. The four benchmark scores remain separate because their task distributions and candidate sets differ.

Table 15: Correct-answer counts, part 1. Seeds are 3407, 7189 and 12011 in order.

Benchmark	Seed 1	Seed 2	Seed 3	Mean (%)
LoRA (LLM)				
ADQA	1487	1499	1498	49.82
VoxP	302	315	310	15.45
MMSU	3112	3137	3123	62.48
MUGEN	547	554	550	31.45
DeepStack injection†				
ADQA	1520	1528	1528	50.84
VoxP	436	442	445	22.05
MMSU	3220	3244	3235	64.66
MUGEN	564	567	567	32.34
PCLM†				
ADQA	1542	1555	1550	51.63
VoxP	546	557	550	27.55
MMSU	3280	3302	3290	65.81
MUGEN	570	578	574	32.80
LoRA (audio+LLM)				
ADQA	1573	1585	1585	52.70
VoxP	614	627	619	31.00
MMSU	3347	3370	3357	67.16
MUGEN	593	602	596	34.11
Projected feedback				
ADQA	1559	1574	1565	52.20
VoxP	490	505	496	24.85
MMSU	3291	3318	3303	66.08
MUGEN	579	587	583	33.31

Denominators: ADQA 3,000; VoxP 2,000; MMSU 5,000; MUGEN 1,750. Counts correspond to the three seeds reported in the main text.

K.1 ADDITIONAL METHODS AND STRONGER CONTROLS

The one-call counts in Table 16 correspond to the early layer-8 configuration. Table 17 separately records the late layer-18 model, the work-controlled joint adapter, and the same-depth interventions. This distinction preserves the training and inference condition associated with each result.

Table 16: Correct-answer counts, part 2. Seeds are 3407, 7189 and 12011 in order.

Benchmark	Seed 1	Seed 2	Seed 3	Mean (%)
AuralLoop, one call at layer 8				
ADQA	1602	1615	1607	53.60
VoxP	607	619	610	30.60
MMSU	3379	3403	3391	67.82
MUGEN	600	607	602	34.46
AuralLoop, two calls				
ADQA	1634	1645	1638	54.63
VoxP	659	670	663	33.20
MMSU	3424	3446	3438	68.72
MUGEN	612	620	616	35.20
LoRA (LLM) + AAD				
ADQA	1520	1531	1515	50.73
VoxP	465	467	473	23.42
MMSU	3172	3145	3165	63.21
MUGEN	543	545	549	31.18
LoRA (LLM) + IAAN				
ADQA	1564	1552	1556	51.91
VoxP	650	659	648	32.62
MMSU	3286	3269	3267	65.48
MUGEN	576	568	573	32.70
Fixed-question conditioning				
ADQA	1584	1585	1599	52.98
VoxP	587	585	593	29.42
MMSU	3359	3362	3383	67.36
MUGEN	599	594	593	34.02

Denominators: ADQA 3,000; VoxP 2,000; MMSU 5,000; MUGEN 1,750. Counts correspond to the three seeds reported in the main text.

Table 17: Per-seed correct counts for late re-entry, training-work control, and same-depth interventions.

Task	N	3407	7189	12011	Mean $\pm$ SD
One re-entry, layer 18					
ADQA	3000	1623	1627	1616	54.07 ± 0.19
VoxParadox	2000	635	650	641	32.10 ± 0.38
MMSU	5000	3408	3418	3421	68.31 ± 0.14
MUGEN	1750	608	603	615	34.78 ± 0.34
Joint LoRA (C)					
ADQA	3000	1602	1619	1611	53.69 ± 0.28
VoxParadox	2000	646	639	649	32.23 ± 0.26
MMSU	5000	3396	3412	3407	68.10 ± 0.16
MUGEN	1750	608	614	611	34.91 ± 0.17
Same-depth shadow, raw					
ADQA	3000	1615	1624	1631	54.11 ± 0.27
VoxParadox	2000	642	638	650	32.17 ± 0.31
MMSU	5000	3412	3428	3419	68.39 ± 0.16
MUGEN	1750	614	609	616	35.03 ± 0.21
Same-depth shadow, norm-matched					
ADQA	3000	1624	1619	1635	54.20 ± 0.27
VoxParadox	2000	647	656	649	32.53 ± 0.24
MMSU	5000	3417	3421	3434	68.48 ± 0.18
MUGEN	1750	615	618	613	35.16 ± 0.14

L ATTRIBUTE AND MULTI-AUDIO RESULTS

The capability breakdowns explain the aggregate results under the shared adaptation protocol. They subdivide the primary benchmarks rather than adding independent test populations.

Table 18: VoxParadox attribute breakdown.

Category	N	LoRA	PCLM	Joint LoRA	AuralLoop
Age	200	8.00	30.17	31.67	34.50
Gender	200	12.00	38.67	40.17	44.33
Emotion	200	9.33	31.50	37.50	41.67
Pitch	200	14.17	17.83	23.17	23.83
Volume	200	13.17	16.50	20.83	21.00
Speed	200	15.00	17.17	20.17	21.83
Range	200	11.50	18.50	22.00	22.17
Intonation	200	15.83	36.00	40.50	48.17
Speaker identity	200	24.67	29.50	33.67	39.50
Speaker count	200	30.83	39.67	40.33	35.00
Overall	2000	15.45	27.55	31.00	33.20

LoRA denotes language-only; Joint LoRA denotes audio-tail plus language updates. All displayed values are three-seed mean accuracies computed from integer correct counts.

Table 19: VoxParadox response allocation: correct answers, adversarial-label agreement, and other errors.

Method	GT accuracy	ALA	Other error
LoRA (LLM)	15.45	66.15	18.40
PCLM†	27.55	56.70	15.75
LoRA (audio+LLM)	31.00	50.42	18.58
AuralLoop, two calls	33.20	47.18	19.62

GT and adversarial labels are distinct. ALA counts selection of the adversarial label; other error covers the remaining responses.

VoxParadox attributes. The attribute comparison tests whether improvement extends across listening skills. Relative to equal-exposure joint LoRA, the largest gain occurs in intonation, followed by speaker identity. Emotion, gender, and age also improve, whereas pitch, volume, and range change little. Because every attribute contains the same number of questions, this uneven pattern is not an artifact of category size. Speaker counting declines even as speaker identity improves. The contrast matters: distinguishing which speaker is present and determining how many speakers are present place different demands on the model. Re-entry benefits the former in this evaluation without providing a general improvement in speaker analysis.

This pattern identifies the acoustic attributes that benefit from re-entry under the tested protocol. Direct encoder adaptation already captures substantial attribute information, while re-entry adds its largest gains on selected distinctions. The breakdown characterizes the resulting capability profile. The fixed-checkpoint interventions in Appendix N separately examine dependence on the feedback path.

Lexical conflict and response allocation. Adversarial-label agreement (ALA) measures selection of the misleading lexical label, whereas ground-truth (GT) accuracy measures selection of the annotated acoustic label (Pang et al., 2026). The three response classes in Table 19 are exhaustive and mutually exclusive, giving $\Delta\mathrm{GT} + \Delta\mathrm{ALA} + \Delta\mathrm{Other} = 0$ between models, where Δ denotes the difference in each response share. This accounting tests whether avoiding the lexical distractor also yields the correct acoustic decision. Against joint LoRA, ALA decreases more than GT accuracy increases, and other errors consequently occupy a larger share. Thus the improvement contains both a net gain in correct answers and a net redistribution among incorrect answers. Reporting GT alone would conceal the latter, while reporting ALA alone would overstate successful acoustic

understanding. These marginal allocations describe the composition of responses. They do not identify transitions among these three response classes. Appendix N separately examines paired changes in correctness.

MMSU perception and reasoning. Table 4 asks whether the improvement extends equally to perceiving speech attributes and reasoning about speech. All three perception categories improve over joint LoRA, with the largest margin in paralinguistic perception. The reasoning categories move in different directions and by much smaller amounts: semantic and phonological reasoning decline slightly, while paralinguistic reasoning increases slightly. This separation is more informative than grouping all speech tasks into one score. It supports an intervention aimed at refining available acoustic evidence, while locating the empirical benefit at the perceptual end of the benchmark. The six category sizes in Table 4 order are 635, 935, 1010, 1108, 977, and 335. Perception contains 2,580 items and reasoning contains 2,420. Aggregate scores weight correct counts by these sizes rather than averaging rounded percentages. This profile describes task performance rather than a measured change in internal representations.

L.1 MULTI-AUDIO COMPARISONS

Table 20: MUGEN dimension breakdown.

Category	N	LoRA	PCLM	Joint LoRA	AuralLoop
Semantics and pragmatics	150	71.11	68.00	70.67	71.33
Speaker and demographics	200	16.17	17.50	19.00	21.33
Affective and paralinguistic	400	34.75	37.25	38.08	39.50
Temporal awareness	250	18.40	20.80	22.67	22.00
Scene and events	250	14.67	16.80	18.53	19.07
Music analysis	250	53.47	53.07	54.00	55.20
Compositional reasoning	250	22.40	24.53	25.07	27.07
Overall	1750	31.45	32.80	34.11	35.20

LoRA denotes language-only; Joint LoRA denotes audio-tail plus language updates. Accuracy values are three-seed means (%).

MUGEN tests whether the QA gains extend to questions about relationships among recordings (Yang et al., 2026b). Its category sizes differ, so the overall score is computed from correct counts rather than an unweighted average of the seven percentages.

Relations among clips. Relative to equal-exposure joint LoRA, speaker and compositional categories show the largest percentage-point gains. The larger affective category also contributes to the overall improvement, while semantics, scene/events, and music show smaller gains. Speaker and affective improvements are consistent with the capability pattern on the speech benchmarks. Compositional improvement adds evidence that revised acoustic information can be useful when the language model must combine evidence across inputs. Each clip is still encoded separately, and each re-entry revises one clip using a condition from the shared language sequence. These results therefore evaluate the utility of the same architecture in a relational setting, without introducing a dedicated multi-audio adaptation corpus.

Temporal awareness. Temporal accuracy declines despite the positive overall result. The distinction between compositional improvement and temporal decline shows that combining information across clips does not improve every type of relation. Together with the VoxParadox speaker-count result, it identifies a specific boundary of the capability gains: better recognition of delivery and identity does not ensure better enumeration or temporal comparison. The reported temporal category remains an aggregate, so it localizes the weakness without attributing it to a particular ordering or alignment operation.

Candidate identity and ordering. The language model receives an ordered collection of clips. A reference retains its identity during candidate rotation, and the gold label follows the selected candidate rather than its original letter. This distinction matters because stable acoustic evidence can

still produce different answers when its position changes. Original-order accuracy, mean accuracy across orders, and remapped-choice consistency measure different behaviors. The category results above address original-order accuracy and do not by themselves establish order robustness. Audio-permutation self-consistency adds inference calls and is evaluated with its own cost, as in the original MUGEN study (Yang et al., 2026b).

M DESIGN SENSITIVITY AND TRAINING-SET SIZE

The extended ablations assess whether additional computation or alternative update rules change the quality–cost tradeoff beyond the main controls.

Table 21: Extended design sensitivity.

Factor	Setting	ADQA	VoxP	MMSU	MUGEN
Re-entry steps	1 (layer 8)	53.60	30.60	67.82	34.46
Re-entry steps	2 (reference)	54.63	33.20	68.72	35.20
Re-entry steps	3	54.73	33.35	68.78	35.26
Anchor depth	Layer 28 / tail 4	53.97	31.72	68.04	34.78
Anchor depth	Layer 20 / tail 12	54.79	33.42	68.83	35.31
Anchor update	Accumulated anchor	54.02	31.92	68.11	34.72
Second revision	$E^2 - E^0$	54.18	32.23	68.22	34.86
Feedback gradient	Stop-gradient	54.20	32.48	68.17	34.78

The default and one-call rows follow the main protocol. Each other row is a separately trained configuration.

Depth and number of calls. The call-count sweep starts from an early single re-entry at layer 8. Adding the second call improves all four benchmarks, whereas a third call produces only small further gains. On VoxParadox, the third call adds 0.15 percentage points while requiring another frozen-tail evaluation. Parameter sharing limits the added trainable weights to a gate pair, but the extra encoder computation remains. This separates parameter efficiency from inference cost when choosing the default. The late single-call comparison in the main text separately tests whether moving one call deeper explains the two-call benefit.

Anchor depth changes both the cached representation and the length of the frozen suffix available after conditioning. Moving the anchor from layer 24 to layer 20 expands the repeated tail from eight to twelve layers, with only small gains on every benchmark. Moving it to layer 28 shortens the repeated tail to four layers but lowers accuracy throughout, most clearly on VoxParadox. Thus reducing acoustic recomputation sacrifices performance, while increasing it beyond the default yields diminishing empirical returns. The development-selected layer-24, two-call configuration retains most of the accuracy of the tested more expensive variants while avoiding their additional tail computation.

The three-call configuration uses language boundaries $(8, 18, 23)$ for 7B and $(12, 24, 30)$ for the 3B engineering configuration. It adds a gate pair and the difference $E^3 - E^2$, while reusing the original anchor and shared weights. The declared anchor/call candidates are selected using internal development data.

Form of the update. The accumulated-anchor variant tests whether the second cell should refine C^1 directly or restart from C^0. Its lower accuracy across all four tasks favors retaining the original acoustic reference as the condition evolves. This comparison concerns where the next acoustic transformation begins. It is distinct from how the resulting embedding is returned to the language stream.

The non-incremental variant tests that return rule by replacing $E^2 - E^1$ with $E^2 - E^0$. Both calls then express changes relative to the initial embedding. The default again leads across all tasks, supporting a second proposal that reports the change from the preceding encoder output. The larger losses on VoxParadox for both update variants make this benchmark particularly sensitive to the choice of reference. These results complement the main feedback interventions by testing the representation update itself.

Stop-gradient preserves the forward conditioning operation but blocks its gradient into the language state. Its consistent accuracy loss supports jointly optimizing the language adapters and cell through the feedback path. Each row is a separately trained configuration under the shared recipe, so these design comparisons include the resulting optimization effects.

M.1 Training-set size

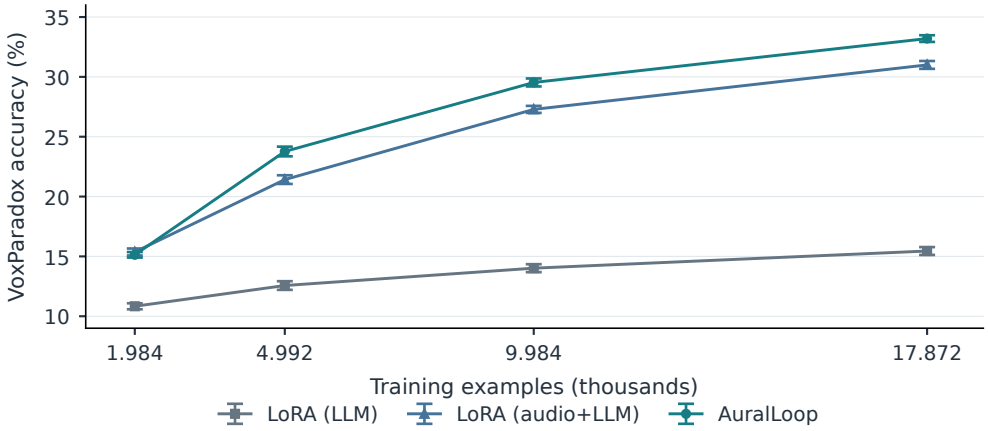

Figure 7: VoxParadox accuracy versus adaptation-set size. Nested source-grouped subsets use two training epochs and the same development set. Error bars denote three-seed standard deviations. Larger subsets also entail more optimizer updates.

The 1,984, 4,992, 9,984, and 17,872 record subsets are nested by source-audio group. The fixed 1,440-record development set preserves the selection criterion as training exposure increases. Figure 7 reports VoxParadox accuracy for language-only LoRA, joint audio–language LoRA, and the default two-call AuralLoop configuration.

All three methods improve as the adaptation set grows, but their relative ordering is informative. At the smallest budget, AuralLoop and joint LoRA have similar means, with joint LoRA slightly ahead. AuralLoop leads from 4,992 examples onward and retains a 2.20-point advantage at the full adaptation size. The gap is comparable across the three larger subsets while both methods continue to improve. The advantage therefore emerges beyond the smallest subset and persists as data increase. Language-only LoRA also benefits from additional examples, yet remains well below both methods that adapt acoustic processing at every budget.

Keeping epochs fixed approximately preserves repeated exposure per example, while increasing the number of distinct source groups and optimizer updates together. The curve consequently measures the practical return from a larger adaptation set under the fixed-epoch recipe. The work-controlled comparison in Appendix H.1 asks the complementary question of whether additional joint-LoRA training on the same data closes the gap. These endpoints characterize attained accuracy, rather than convergence speed or checkpoint stability. This sweep uses the 7B backbone. The 3B configuration elsewhere serves development and engineering purposes.

N Paired decisions and feedback interventions

Paired decisions distinguish net accuracy gains from changes in individual answers. Table 22 retains the comparison between independently trained early one-call and two-call models. Correct/correct and wrong/wrong cells indicate agreement in correctness. Wrong/correct and correct/wrong cells measure repairs and regressions.

Table 22: Paired correctness counts for separately trained early one-call and two-call models.

Benchmark	Seed	C/C	W/C	C/W	W/W
ADQA	3407	1541	93	61	1305
ADQA	7189	1551	94	64	1291
ADQA	12011	1545	93	62	1300
VoxP	3407	586	73	21	1320
VoxP	7189	595	75	24	1306
VoxP	12011	588	75	22	1315
MMSU	3407	3304	120	75	1501
MMSU	7189	3321	125	82	1472
MMSU	12011	3313	125	78	1484
MUGEN	3407	541	71	59	1079
MUGEN	7189	546	74	61	1069
MUGEN	12011	542	74	60	1074

The first label refers to the one-call model; the second to the two-call model. Each row sums to its benchmark size and matches both marginal correct counts. Each row uses the same evaluation items for both models.

On MUGEN, the early-call comparison averages 73 repairs and 60 regressions, leaving 13 additional correct answers. Its modest net improvement therefore coexists with substantially more changed outcomes. These are comparisons between trained configurations, not successive decoded answers in a single inference trajectory.

N.1 Late single-call comparison

Table 23: Paired correctness counts for the late single-call model (layer 18) and the two-call model. The first correctness label refers to the late single-call model.

Task	Seed	C/C	W/C	C/W	W/W
ADQA	3407	1571	63	52	1314
ADQA	7189	1573	72	54	1301
ADQA	12011	1565	73	51	1311
VoxParadox	3407	602	57	33	1308
VoxParadox	7189	613	57	37	1293
VoxParadox	12011	604	59	37	1300
MMSU	3407	3319	105	89	1487
MMSU	7189	3331	115	87	1467
MMSU	12011	3333	105	88	1474
MUGEN	3407	576	36	32	1106
MUGEN	7189	569	51	34	1096
MUGEN	12011	586	30	29	1105

Rows sum to the benchmark size. C/C + C/W recovers the late-call correct count; C/C + W/C recovers the two-call count.

Table 23 supplies the joint counts behind Figure 4. The stronger layer-18 single-call model reduces the net gap relative to the early comparison, while still leaving more repairs than regressions in each benchmark's aggregate.

The early comparison tests adding a second re-entry after a language interval that has processed the first revision. The late comparison asks whether placing a single call at the deeper boundary already captures that benefit. The smaller gap is consistent with call placement contributing to the early comparison's advantage. The remaining repairs and regressions describe changes in the decision boundary between independently adapted models. They do not imply that a model first generated an incorrect answer and then corrected it within one run.

N.2 SAME-CHECKPOINT INTERVENTIONS

Table 24: Feedback-content interventions on the same two-step checkpoint, without retraining.

Condition	ADQA	VoxP	MMSU	MUGEN
Intact feedback	**54.63**	**33.20**	**68.72**	**35.20**
Zero second injected residual	53.90	30.90	67.93	34.63
Reuse r^1 at step 2	53.67	30.97	67.93	34.51
Shuffle r^2 only	53.18	30.73	67.37	34.17
Norm-matched donor second residual	53.54	30.58	67.65	34.34
Same-depth shadow	54.11	32.17	68.39	35.03
Same-depth shadow, norm-matched	54.20	32.53	68.48	35.16
Shuffle both r^1 and r^2	51.46	27.18	65.76	32.93

Weights and step gates stay fixed. Norm matching controls the injected residual magnitude; the receiving first revision is retained except when both conditions are shuffled.

All interventions retain the two-call checkpoint's weights, step gates, complete language stack, final normalization, and output head. Second-step interventions execute the intact first revision, then change only the second condition or injected residual. The both-condition shuffle also replaces r^1 and recomputes the first revision. Both shuffled conditions come from the same donor's intact path, precomputed with the corresponding seed's two-call checkpoint. The receiver recomputes its first revision from donor r^1, then its second from donor r^2. Suppressing the second residual in this checkpoint differs from independently training a one-call model.

Suppression, reuse, and donor conditions. Setting the second injected residual to zero measures dependence on that path. Reusing r^1 removes condition refresh, while potentially collapsing the second difference when the anchor gates agree. Second-step donor replacement substitutes another sample's condition. Donors are matched within joint clip-count and duration buckets, excluding the receiver and its source-audio component. Duration is the sum of all valid clip durations in the question, including a reference when present. The half-open buckets are $[0, 10)$, $[10, 30)$, $[30, 60)$, $[60, 180)$, $[180, 300)$, and $[300, \infty)$ seconds. Stable sample-ID ordering and seed 9021 determine candidate order for a perfect matching. A bucket without a feasible matching uses intact feedback for its receivers and retains them in the full evaluation denominator. This assignment fallback is distinct from the near-zero proposal fallback below. The receiving audio, question, first revision, labels, gates, and weights are unchanged.

Let $U_i^{(2)} \in \mathbb{R}^{T_{a,i} \times d_\ell}$ be the intact injected residual for clip i, including the reference scale and channel gate. Only the donor conditioning vector is transferred. The receiver's cell and tail process its own C_i^0 and μ_i, then subtract its intact E_i^1 to form $\widetilde{U}_i^{(2)}$. The proposal therefore matches the receiver's acoustic length even if the donor has a different length. The amplitude-controlled intervention is

$$U_{i,\text{donor}}^{(2)} = \begin{cases} \dfrac{\|U_i^{(2)}\|_F}{\|\widetilde{U}_i^{(2)}\|_F} \widetilde{U}_i^{(2)}, & \|\widetilde{U}_i^{(2)}\|_F > \epsilon_d, \\ 0, & \text{otherwise}, \end{cases} \qquad \epsilon_d = 10^{-8}. \tag{15}$$

Frobenius norms are computed in FP32 over valid positions, separately for each clip. For receivers with a valid donor assignment, near-zero donor-conditioned proposals receive the explicit zero fallback and remain in the evaluation denominator. Unmatched receivers bypass donor substitution and norm matching entirely and use the intact path. Matching the whole-clip norm controls overall amplitude, while allowing the temporal and channel distribution to change. The intact residual norm is available to this diagnostic, not to a separate deployment controller.

Assignment coverage. The assignment is shared across the second-condition shuffle, norm-matched donor, and both-condition shuffle, and across training seeds. The numbers of receivers reverting to intact feedback are 9 of 3,000 on ADQA, 4 of 2,000 on VoxParadox, 9 of 5,000 on MMSU, and 7 of 1,750 on MUGEN, per seed and per diagnostic. Thus the fractions with a valid donor assignment are 99.70%, 99.80%, 99.82%, and 99.60%, respectively. These counts describe matching support,

separately from the activation-dependent near-zero proposal fallback in Equation (15). Each seed computes its own donor condition vectors using the shared assignment.

Let $\mathcal{S}$ be the successfully matched receivers in a benchmark of size N. Receivers outside $\mathcal{S}$ use the intact path, so their correctness differences are zero. For the same fixed checkpoint and deterministic evaluator,

$$\Delta_{\text{full}} = \frac{|\mathcal{S}|}{N}\Delta_{\mathcal{S}}, \tag{16}$$

where both Δ_{full} and $\Delta_{\mathcal{S}}$ denote intact-minus-intervention accuracy differences in percentage points, on the full set and matched subset, respectively. Reported tables use the full denominator. The high matching coverage limits dilution from unsupported buckets, while a valid assignment alone does not guarantee a nonzero or behavior-changing residual.

Table 25: Seed-level correct counts for interventions on a fixed two-call checkpoint.

Condition	Test	3407	7189	12011	Mean
Zero second injected residual	ADQA	1611	1623	1617	53.90
	VoxP	612	625	617	30.90
	MMSU	3385	3407	3397	67.93
	MUGEN	604	609	605	34.63
Norm-matched donor second residual	ADQA	1601	1611	1607	53.54
	VoxP	606	618	611	30.58
	MMSU	3371	3394	3383	67.65
	MUGEN	596	605	602	34.34

The intervention pattern distinguishes dependence on the second update from sensitivity to its condition. Suppression lowers accuracy on every benchmark, showing that the trained model uses this path. Reuse performs similarly to suppression on VoxParadox and MMSU, but also changes conditioning depth and can shrink the incremental residual. A donor residual remains harmful after its norm is matched, and performs worse than suppression on all four benchmarks. Thus a residual of the expected overall magnitude is insufficient: the tested donor-conditioned update can disrupt decisions more than omitting the update. Shuffling both conditions causes the largest loss, consistent with sensitivity to conditioning across the re-entry sequence.

N.3 Same-depth condition without the first feedback

Reusing r^1 changes both language depth and exposure to the first revision. A same-depth control instead computes a shadow condition from the pre-injection state,

$$\bar{H}^{2,\text{sh}} = D^{\omega,\eta}_{b_1+1:b_2}(\bar{H}^1), \qquad r^{2,\text{sh}} = (\bar{H}^{2,\text{sh}}_{p,:})^\top. \tag{17}$$

The shadow branch uses the same middle language layers as the intact second stage, but receives $\bar{H}^1$ rather than H^1. It therefore retains the original audio and question while excluding the first acoustic injection. The main receiving path still contains its intact first revision.

The alternate encoder output and second residual are

$$E^{2,\text{sh}}_i = T_\phi\big(F_\theta(C^0_i, r^{2,\text{sh}}; \alpha_2); \mu_i\big)\,, \tag{18}$$

$$(U^{(2)}_{i,\text{sh}})_{j,:} = \left[\gamma_2 \odot \frac{e^{2,\text{sh}}_{i,j} - e^1_{i,j}}{s_{i,j}}\right]^\top. \tag{19}$$

Here $e^{2,\text{sh}}_{i,j} = (E^{2,\text{sh}}_{i,j,:})^\top$. The subtraction reference is the receiver's intact E^1_i. The norm-matched version applies Equation (15) to this shadow proposal, matching the intact second residual's norm per clip. Both variants run in evaluation mode without retraining and use the unchanged final output path.

The same-depth, norm-matched intervention reduces VoxParadox accuracy by 0.67 points, but changes MUGEN by only 0.04 points. Thus sensitivity to refreshed language conditioning depends on the task. The intervention removes a specific feedback dependency while controlling depth and

overall residual amplitude. It can still shift the hidden-state distribution. Its extra middle-stage and tail computations belong to the diagnostic cost rather than ordinary serving cost.

Depth and amplitude controls address different explanations. The shadow condition retains the later language depth. Rescaling a nonzero shadow residual to the intact norm separates its overall size from its temporal and channel pattern. VoxParadox remains sensitive after both controls, supporting a role for language states that have incorporated the first acoustic revision. MUGEN responds differently: removing the second update hurts performance, yet replacing its condition with the norm-matched shadow changes little. The small MUGEN mean difference does not establish equivalence. It indicates weaker observed sensitivity to this feedback dependency than to removal of the second update. Thus aggregate multi-audio gains need not entail the same conditioning sensitivity as paralinguistic gains.

N.4 Paired training-seed margins

For benchmark B of size N_B, let $c_{A,s,B}$ and $c_{R,s,B}$ be correct counts for two-call AuralLoop A and reference R at seed s. Their paired margin is

$$d_{s,B} = 100(c_{A,s,B} - c_{R,s,B})/N_B, \qquad \bar{d}_B = \frac{1}{3}\sum_{s=1}^{3} d_{s,B}. \tag{20}$$

The margin SD is $s_{d,B} = \sqrt{\frac{1}{2}\sum_{s=1}^{3}(d_{s,B} - \bar{d}_B)^2}$, computed from the three paired differences. It is not the difference of two method SDs. Matched seed identifiers align the experiment but do not make distinct architectures' parameter draws identical.

Table 26: Per-seed accuracy margins of two-call AuralLoop in percentage points.

Benchmark	3407	7189	12011	Mean ± SD
Relative to equal-exposure joint LoRA				
ADQA	+2.03	+2.00	+1.77	$+1.93 \pm 0.15$
VoxParadox	+2.25	+2.15	+2.20	$+2.20 \pm 0.05$
MMSU	+1.54	+1.52	+1.62	$+1.56 \pm 0.05$
MUGEN	+1.09	+1.03	+1.14	$+1.09 \pm 0.06$
Relative to early one-call re-entry				
ADQA	+1.07	+1.00	+1.03	$+1.03 \pm 0.03$
VoxParadox	+2.60	+2.55	+2.65	$+2.60 \pm 0.05$
MMSU	+0.90	+0.86	+0.94	$+0.90 \pm 0.04$
MUGEN	+0.69	+0.74	+0.80	$+0.74 \pm 0.06$
Relative to late one-call re-entry				
ADQA	+0.37	+0.60	+0.73	$+0.57 \pm 0.19$
VoxParadox	+1.20	+1.00	+1.10	$+1.10 \pm 0.10$
MMSU	+0.32	+0.56	+0.34	$+0.41 \pm 0.13$
MUGEN	+0.23	+0.97	+0.06	$+0.42 \pm 0.49$
Relative to work-controlled joint LoRA				
ADQA	+1.07	+0.87	+0.90	$+0.94 \pm 0.11$
VoxParadox	+0.65	+1.55	+0.70	$+0.97 \pm 0.51$
MMSU	+0.56	+0.68	+0.62	$+0.62 \pm 0.06$
MUGEN	+0.23	+0.34	+0.29	$+0.29 \pm 0.06$

The stronger controls narrow the margins and change their variability. In particular, the small mean MUGEN margin over the late one-call model varies substantially across seeds. These SDs describe training-seed variation, not uncertainty from sampling the audio population.

All displayed seed-paired margins are positive, so the observed direction is consistent across the three runs for each benchmark and reference. Stability of magnitude is a separate property. VoxParadox's advantage over late single re-entry varies little, whereas its work-controlled margin is more sensitive to the training run. MUGEN makes this distinction especially clear: its late-call margin ranges from 0.06 to 0.97 points, although none of the runs reverses the ranking. The paired table therefore shows a consistent ordering in these runs and distinguishes effect sizes that vary little from those more sensitive to the training seed.

O STATISTICAL ESTIMANDS

For $S = 3$ training seeds and N benchmark items, let $z^R_{s,i}, z^A_{s,i} \in \{0, 1\}$ denote correctness for reference R and two-call AuralLoop A. Figure 4 uses the late single-call model as R. The rates average over the same item–seed pairs:

$$R_{\text{fix}} = \frac{1}{SN} \sum_{s=1}^{S} \sum_{i=1}^{N} \mathbf{1}[z^R_{s,i} = 0, z^A_{s,i} = 1],$$
$$R_{\text{break}} = \frac{1}{SN} \sum_{s=1}^{S} \sum_{i=1}^{N} \mathbf{1}[z^R_{s,i} = 1, z^A_{s,i} = 0]. \tag{21}$$

Here $\mathbf{1}[\cdot]$ is the indicator function. The difference in three-seed mean accuracy, in percentage points, is exactly $100(R_{\text{fix}} - R_{\text{break}})$. Wrong/wrong pairs contribute to neither rate, even when the predicted labels differ. Repair and regression counts therefore explain changes in correctness, rather than all answer changes.

Training variability and source dependence. Reported means and sample SDs summarize three independently trained checkpoints. The paired seed margins in Table 26 quantify how each method contrast varies across these runs. Questions can share an original recording, overlapping crops, or a duplicate source. The tables therefore describe performance on the evaluated benchmark populations and training-seed variation, without confidence intervals or significance claims about resampling the underlying audio population.

Comparison scope. The principal contrasts are two-call AuralLoop against direct joint adaptation, work-controlled joint adaptation, and late single re-entry. Static-feature access and fixed-checkpoint interventions address complementary questions. Attribute and multi-audio breakdowns describe where each aggregate effect arises. They are not independent replications of the overall comparison.

Failures and denominators. Strict accuracy counts invalid answers and runtime failures as incorrect in the full benchmark denominator. Format errors, timeouts, unsupported inputs, and out-of-memory failures retain separate status fields. A paired comparison uses the same sample IDs, candidate mappings, and gold labels for both methods. Thus neither a change in answer format nor a runtime failure silently changes the evaluated population.

P LATENCY AND MEMORY ACCOUNTING

Profiling uses the same GPU topology, model sharding, precision, attention backend, and output policy across methods. Time to first token includes online preprocessing, transfer, native acoustic encoding, re-entry, language prefill, and first-token selection. Decoder-only token latency measures a different portion of the request.

Deployment configuration. All linearly mergeable audio and language LoRA weights are merged at the common deployment precision. Each static branch's low-rank residual can likewise be folded into its corresponding dense projection. The PCLM prompt-dependent mixer, DeepStack's branches and gates, and AuralLoop's cell and repeated encoder tail remain active. Each system retains the Thinker, audio encoder, and required projector, with the unused Talker and token-to-waveform modules disabled. The same sharding and cache policies apply across methods.

Table 27: Trainable parameters, time to first token, and LibriSpeech word error rate.

Method	Trainable (M)	TTFT ×	LS-clean	LS-other
Qwen2.5-Omni (native)	0.000	1.000	1.81	3.44
LoRA (LLM)	3.211	1.009	1.77	3.32
LoRA (audio+LLM)	3.244	1.012	1.83	3.48
AuralLoop, one call (layer 8)	3.192	1.084	1.83	3.47
AuralLoop	3.196	1.162	1.86	3.53

TTFT is normalized to native Qwen on the 60-s workload. LS-clean and LS-other report WER (lower is better). Trainable counts include active adapters and step gates.

Parameter efficiency and transcription quality. Table 27 separates trainable storage, prompt-processing cost, and transcription quality. The language-only and joint adapters have similar parameter budgets to AuralLoop, yet their TTFT remains close to the native backbone. The difference follows from the computation performed by the added interface. Re-entry shares tail weights across calls but executes the tail again, so a compact adapter need not be a low-latency adapter.

LibriSpeech WER rises slightly for the two-call model on both test sets, with the larger absolute change on test-other. The QA improvements therefore coexist with a small transcription regression. This result places the adaptation's benefit in acoustic question answering rather than general transcription improvement. Frozen pretrained weights preserve the original parameters, while the added conditioning and injection paths can still change the model's behavior.

Transcription scoring. References and hypotheses use the same AL-EN-v1 normalization and ASR prompt specified in Appendix J. Word sequences are aligned by unit-cost Levenshtein edit distance, with a cost of one for each substitution, deletion, or insertion. Corpus WER is $100(\sum S + \sum D + \sum I)/\sum N_{\text{ref}}$, where S, D, I are utterance-level edit counts and N_{ref} is the reference word count. This aggregates word errors across the corpus rather than averaging per-utterance percentages.

Table 28: **Marginal accuracy and prefill cost.** L is the layer-18 single-call model. C is the additional-training-work joint adapter. TTFT ratios compare mean times on the same QA requests.

	Accuracy gain (pp)		Mean TTFT ratio	
Benchmark	vs. L	vs. C	Two/L	Two/C
ADQA	+0.57	+0.94	1.089	1.204
VoxParadox	+1.10	+0.97	1.084	1.193
MMSU	+0.41	+0.62	1.087	1.199
MUGEN	+0.42	+0.29	1.100	1.234

Ratios are ratios of mean request times, including acoustic encoding and re-entry. Appendix P gives the timing boundary and workload definitions.

Comparing late re-entry and work-controlled joint adaptation. Table 28 reports latency ratios on the same QA requests for each comparator. Relative to the late single-call model, two calls trade an additional prefill increment for staged acoustic revision. Relative to Joint LoRA (C), the latency increase is larger because the joint adapter spends its extra computational allowance during training. These are different deployment choices even when their training resources are controlled. VoxParadox combines the largest late-call accuracy margin with the smallest relative TTFT increment among the four QA workloads. MUGEN has a smaller margin and a larger increment, making its aggregate tradeoff less favorable. The duration-controlled experiment below examines a separate standardized workload.

QA-request timing and aggregation. Table 28 uses all 3,000 ADQA, 2,000 VoxParadox, 5,000 MMSU, and 1,750 MUGEN questions, preserving the evaluation's audio and candidate order. For every question and training seed, one complete greedy QA response supplies the accuracy measurement, and five separate calls time the identical prefill through first-token selection. Later punctuation or

end-token decoding is excluded from TTFT. For benchmark size N, the mean time for method M is

$$\bar{t}_M = \frac{1}{3N} \sum_{s \in \{3407,7189,12011\}} \sum_{i=1}^{N} \frac{1}{5} \sum_{r=1}^{5} t_{M,s,i,r}. \tag{22}$$

Here $t_{M,s,i,r}$ is the recorded first-token time for method M, training seed s, question i, and repeat r. Each reported ratio is $\bar{t}_{\text{two}}/\bar{t}_{\text{reference}}$, not the mean of per-request ratios.

QA timing coverage. The four benchmarks contain 11,750 questions, yielding 35,250 question/seed pairs and 176,250 scheduled timing calls per method. Two-call AuralLoop, the late single-call model, and Joint LoRA (C) each produced a first token and valid TTFT in every scheduled call. All three methods used identical question, seed, and repeat indices, with no missing records or retry replacements. Equation (22) therefore uses the complete, common denominators, without zero-filled timings or excluded requests. Repeated timing calls do not create additional independent QA examples.

Duration-controlled protocol. The duration buckets are 10, 30, 60, 180, and 300 seconds. A separate systems-recording pool, disjoint from adaptation sources and unused for model selection, supplies 20 fixed continuous windows per bucket. Each window starts at a fixed offset within a sufficiently long recording. Long windows are neither constructed by repeating short clips nor treated as long-form QA examples. Each window receives five warm-up calls and five recorded calls, giving 100 recorded calls per bucket. Profiling uses the development-selected checkpoint for training seed 3407. The complete rendered systems prompt contains 128 non-audio tokens, including native role and generation-prefix tokens. Batch size is one, and one token is generated for TTFT. All participating GPUs are synchronized around timed calls. Methods use identical window order with rotated model order.

Timing boundary and cache policy. Waveform reading and decoding precede the timed interval, which starts at CPU feature extraction. Processor-only and GPU-prefill times are additional views of the same request, rather than quantities added twice. The primary regime reuses neither the acoustic anchor nor KV states across requests.

Relative latency. Native TTFT is normalized independently within each duration bucket. At 60 seconds, early one-call, two-call, and three-call ratios are 1.084, 1.162, and 1.221. These ratios describe the specified workload and do not determine absolute latency or the duration distribution of a benchmark. The QA-request timings in Table 28 provide the corresponding benchmark-specific comparisons.

Memory and failures. Peak allocated and reserved memory, maximum per-device usage, and summed device usage are separate metrics. Counters are reset before each recorded call. The analytic anchor payload does not include all live activations or distributed allocations. Unsupported context lengths and out-of-memory cases retain their failure status rather than receiving interpolated successful timings. An altered sharding configuration constitutes a separate profiling condition.

P.1 Duration, component means, and latency quantiles

Table 29: Ratio of mean TTFTs by input duration, normalized to the native Qwen mean within each row. Each mean uses 20 windows and five recorded repeats at seed 3407.

Duration (s)	Native	Joint LoRA	Early one call	Two calls
10	1.000	1.008	1.056	1.102
30	1.000	1.010	1.071	1.131
60	1.000	1.012	1.084	1.162
180	1.000	1.016	1.093	1.187
300	1.000	1.019	1.102	1.213

How duration changes the cost. The relative overhead grows with recording length for both re-entry configurations (Table 29). Longer inputs expose more acoustic positions to the repeated tail, consistent with the added encoder computation in Figure 2. Direct joint adaptation stays much closer to the native cost throughout the tested durations. The two-call overhead is therefore not a fixed request-level surcharge. Because each duration uses its own native denominator, the table characterizes relative overhead within a bucket and does not provide an absolute latency curve across buckets.

Table 30: Mean-time components for the 60-second workload, in native-relative units.

Method	Host	Audio	LLM/other	Entry 1	Entry 2	Total
Native	0.117	0.263	0.620	0.000	0.000	1.000
Joint LoRA	0.117	0.275	0.620	0.000	0.000	1.012
Early one call	0.117	0.263	0.620	0.084	0.000	1.084
Two calls	0.117	0.263	0.620	0.084	0.078	1.162

Where the additional time is spent. The component means attribute the added time to the two re-entry operations (Table 30). The initial audio pass remains part of every AuralLoop request, followed by the first and second tail calls. Their costs are similar, as expected from reusing the same tail and acoustic lengths, but weight sharing does not remove either execution. The second revision's modest accuracy increment must therefore be assessed against its additional work rather than against trainable-parameter growth alone.

The component intervals are nonoverlapping portions of the same recorded request, so their means add to the mean total. Every entry in Table 30 divides a component mean by the same native mean TTFT for the 60-second workload. Component medians and quantiles generally cannot be added. Total request time is also recorded directly. Extra donor and shadow computations belong to intervention diagnostics and are excluded from ordinary serving comparisons.

Table 31: Distribution of paired request-level TTFT ratios at 60 seconds. P95 is the 95th percentile of these ratios.

Method	Recorded calls	Median	P95
Native	100	1.000	1.000
Joint LoRA	100	1.012	1.020
Early one call	100	1.084	1.116
Two calls	100	1.162	1.213

Let $t_M(i)$ be method M's TTFT on recorded call i, with 0 denoting native Qwen. Calls are paired by the same source window and repeat index, at the fixed profiling seed 3407, yielding 100 ratios per method. The P95 of paired ratios $t_M(i)/t_0(i)$ is distinct from the ratio of marginal P95 times and from the mean-time ratios above. We use linear interpolation, also called the type-7 quantile. For the sorted 100 paired ratios $r_{(1)} \le \cdots \le r_{(100)}$, P95 is $0.95r_{(95)} + 0.05r_{(96)}$. Native ratios divide each saved native time by itself and therefore equal one, while the underlying native runtimes can vary across calls. P95 characterizes the upper tail of these ratios, not uncertainty in the estimated mean.

Typical and high-overhead requests. Table 31 shows that the upper tail of the paired overhead distribution lies above its median for every adapted configuration. The separation is greater for two calls than for direct joint adaptation. A mean-time comparison alone therefore omits the additional overhead encountered by the higher-ratio requests. Together with the duration experiment, these results support reporting workload-specific latency alongside QA accuracy, especially when choosing between a late single call and two staged calls.

Q External checkpoints and source-reported results

Q.1 Released checkpoints evaluated locally

Table 32: Released checkpoints evaluated under a common label-correctness metric, with format-specific extraction described in Appendix Q. Their training histories differ.

Method	ADQA	VoxParadox	MMSU
Qwen2.5-Omni-7B (Xu et al., 2025)	45.17	7.65	60.54
MOSS-Audio-8B-Thinking (Yang et al., 2026a)	56.77	28.85	75.88
AudioMCQ Mixed-to-Strong (He et al., 2026a)	52.37	19.70	68.34
Qwen2-Audio + PCLM + DPO (Pang et al., 2026)	51.63	71.65	65.92

Table 32 evaluates complete checkpoints with their pretrained audio interfaces and chat serialization. The local QA instructions and output extraction are specified below. Their original training histories are unchanged.

Checkpoint identities. The evaluated external weights are `OpenMOSS-Team/MOSS-Audio-8B-Thinking`, `inclusionAI/AudioMCQ-Mixed-To-Strong`, and `IHP-Lab/Qwen2-Audio_PCLM_DPO`. The last is the released Qwen2-Audio PCLM+DPO system, distinct from the parameter-budgeted PCLM port in Appendix I. It requires the authors' custom `Qwen2AudioForConditionalGeneration` implementation from `qwen2audio.modeling_qwen2audio`, with its PCLM modules enabled and the processor tokenizer passed to `set_llm_tokenizer`.

Generation allowances and extraction. Native Qwen2.5-Omni-7B, AudioMCQ Mixed-to-Strong, and Qwen2-Audio+PCLM+DPO receive label-only QA instructions, generate at most 16 new tokens, and are scored by the whole-response parser in Appendix J. MOSS-Audio-8B-Thinking retains its native reasoning format, with a local 4,096-new-token allowance. Our evaluator extracts a label only from the answer segment after reasoning: the last complete line beginning with `Answer:`, followed by one valid label and an optional period, supplies the candidate answer. Missing final-answer segments, conflicting answer lines, truncated outputs, and runtime failures count as incorrect. Earlier reasoning text is not scanned for a label. Thus, the reasoning checkpoint uses the same label-correctness metric but a different response extractor from the shared-backbone strict-label track. The local instructions request plain option labels rather than AudioMCQ's released answer tags or the PCLM evaluator's label-plus-choice-text responses. These complete-system comparisons do not control training data, generation budget, or training computation.

Q.2 Source-reported measurements

Table 33 reports values from the cited sources rather than the local evaluations above. It provides external context while preserving differences in model training, benchmark release, and scoring protocol.

Table 33: Source-reported results under the original evaluation protocols.

Benchmark	System and source	Score
ADQA final eval	Tathe_UIUC_task5_1 (DCASE Task 5 Organizers, 2026)	50.60
ADQA final eval	Lim_CAU_task5_1 (DCASE Task 5 Organizers, 2026)	57.30
VoxParadox	Qwen2.5-Omni, IAAN reproduction (Huang et al., 2026)	7.60
VoxParadox	Qwen2.5-Omni + IAAN (Huang et al., 2026)	29.00
VoxParadox	Qwen2-Audio + PCLM + DPO (Pang et al., 2026)	72.30
VoxParadox	Audio Flamingo 3 + PCLM + DPO (Pang et al., 2026)	65.20
MMSU (47-task)	Qwen2.5-Omni-7B (Wang et al., 2026)	60.57
MUGEN	Qwen2.5-Omni (Yang et al., 2026b)	28.69
MUGEN	Qwen2.5-Omni + APSC + CoT (Yang et al., 2026b)	31.31

System identity. DCASE identifiers denote complete submitted systems, whose scores are not native-checkpoint accuracies. Source PCLM+DPO scores retain their backbone, supervised training, and preference optimization. Our controlled port instead evaluates question-conditioned mixing under the shared adaptation recipe.

Scoring differences. MUGEN's source protocol includes output handling and audio-permutation strategies that differ from a single strict-label pass. The VoiceBench-related MMSU evaluation in SALAD differs from the 47-task suite used here (Cuervo et al., 2026). Keeping source values separate avoids attributing evaluator or training differences to the re-entry architecture.

R LIMITATIONS

Representation and conditioning. Re-entry transforms information retained by the anchor and its metadata. An earlier anchor can preserve different detail at greater cost, while the low-rank cell limits the available transformations. The condition r^k is a hidden state rather than a decoded reasoning step. Multiplicative conditioning can still use lexical shortcuts, and norm-matched interventions can change the hidden-state distribution. The same-depth controls support task-dependent effects rather than uniform semantic refinement.

Input and serving regime. AuralLoop assumes a complete observed question before feedback begins. It therefore addresses offline prompt prefill rather than streaming question arrival or incremental multi-turn audio dialogue. Upper-layer acoustic KV states are question-dependent and cannot be reused unchanged across new questions. Two calls also introduce additional prefill work even for examples that do not benefit.

Evaluation scope. The reported adaptations use Qwen2.5-Omni-7B and a compact answer-only training resource. The 3B configuration serves engineering purposes rather than cross-scale performance evidence. The budget-constrained mechanism ports do not reproduce complete external pretraining or preference-optimization pipelines. Attribute gains coexist with regressions in speaker counting and temporal awareness. Strict label accuracy also differs from open-ended answer quality, and source overlap limits the independence of benchmark findings.

Quality and cost. Additional re-entry improves mean acoustic QA accuracy, but does not dominate every category or every cost-sensitive operating point. The stronger late-call and work-controlled baselines narrow its advantage. Paired decisions and first-token latency make this tradeoff explicit. Adaptive stopping, broader backbone transfer, and streaming operation are useful extensions beyond the evaluated architecture.