

LEAF: Growing Trees Without Branching for Speech-Aware Large Language Model Post-Training

Argyrios Gerogiannis*
ag91@illinois.edu

Yekaterina Yegorova
yay2@illinois.edu

Mark Hasegawa-Johnson
jhasegaw@illinois.edu

Venugopal V. Veeravalli
vvv@illinois.edu

The Grainger College of Engineering
University of Illinois Urbana-Champaign

Abstract

State-of-the-art GRPO-style methods for speech-aware large language model post-training suffer from coarse credit assignment, broadcasting the same terminal-reward advantage to every token in a response. This ignores useful structure within rollout batches, where speech-conditioned completions often share prefixes before diverging at important decisions. We propose Low-rank Exploration with Adaptive Forking (LEAF), a retrospective tree-based RL method that recovers this structure without online branching or additional decoding. LEAF samples complete responses, selects high-surprisal boundaries, groups responses by shared prefixes, and assigns span-level advantages using descendant rewards. We theoretically justify LEAF’s span-level credit assignment and boundary-selection design. Empirically, LEAF improves over GRPO across speech question answering and speech translation benchmarks under the same rollout and low-rank adaptation budget. Notably, smaller LEAF-trained models consistently outperform current state-of-art, full-parameter baselines.

1 Introduction

Speech-aware large language models (SALLMs) (Arora et al., 2025) enable end-to-end spoken interaction across multiple tasks such as spoken question answering (SQA), automatic speech translation (AST), and speech recognition. As these models improve, post-training becomes essential for aligning generation with task-level objectives not captured by next-token prediction alone. Reinforcement learning (RL) is a natural fit, and recent GRPO-style methods (Shao et al., 2024; Yu et al., 2025) have shown that reward-driven adaptation can improve open-form speech understanding while avoiding additional training costs by normalizing terminal rewards across multiple completions.

Despite its simplicity, GRPO is limited by coarse credit assignment: it computes a terminal reward per response and broadcasts an identical scalar advantage to every token. While efficient, this discards *where* a completion becomes promising or harmful. The issue is especially pronounced in open-form speech post-training, where candidate answers often share early segments and diverge only around uncertain semantic decisions, so the bottleneck is not reward design alone but *rollout organization* and *credit assignment*.

Tree-structured RL offers a finer alternative: by branching from intermediate states and using descendant outcomes for process-level supervision, it localizes credit to the decisions that matter. Directly importing online tree expansion into SALLM post-training is costly however, since speech inputs are long and multimodal, decoding is batched, and rewards are task-specific. This motivates our central question: *can we obtain the benefits of structured exploration and process-aware credit assignment while retaining the simplicity of batched GRPO-style training for SALLMs?*

We answer with **Low-rank Exploration with Adaptive Forking (LEAF)**, a retroactive RL method for SALLM post-training. LEAF samples the same K complete responses as GRPO, then reorganizes them into a sparse retrospective prefix tree. It selects a small number of high-surprisal token boundaries, groups completions by exact shared prefixes at those boundaries, estimates prefix values from descendant terminal rewards, and assigns span-level advantages between retained nodes. LEAF therefore keeps GRPO’s batched generation pattern while replacing sequence-level credit with structured span-level supervision derived from the rollout batch itself. Finally, LEAF is designed for parameter-efficient speech post-training. We instantiate it with LoRA (Hu et al., 2022), motivated by recent evidence that low-rank adaptation can approach full-parameter RL post-

*Corresponding author.

training performance (Schulman and Lab, 2025). This lets us test whether better credit assignment can translate into stronger SALLM adaptation under a lightweight trainable-parameter budget. To our knowledge, LEAF is the first tree-based RL method for speech post-training, and the first post-training method within retrospective credit assignment to use an uncertainty-guided mechanism for constructing prefix-level supervision.

Contributions. We introduce LEAF, a retroactive tree-based RL method for SALLM post-training that extracts process-aware span advantages from the same i.i.d. rollout groups used by GRPO. We provide theoretical support for LEAF’s design, showing that span-level prefix advantages are valid credit signals, that the fork budget controls a support-resolution tradeoff, and that pure prefix matching is biased toward shallow high-collision prefixes. Empirically, we show that speech-understanding rollouts are highly forkable across tasks and backbones, and that LEAF consistently improves over GRPO under the same rollout and LoRA adaptation budget, with gains in automatic metrics, judge scores and tail-risk behavior. The implementation of LEAF is available at: <https://github.com/Grimoid/LEAF>.

2 Related Work

GRPO for SALLMs. GRPO (Shao et al., 2024) is a critic-free RL objective that normalizes terminal rewards within a group of sampled completions. Recent work applies GRPO to SALLMs for open-form speech understanding, including SQA and AST (Elmakies et al., 2026), as well as to automatic speech recognition (Shivakumar et al., 2025) and text-to-speech (Liu et al., 2026). These works show that reward-driven post-training is effective for speech, but retain GRPO’s flat credit assignment: one terminal-reward advantage is broadcast to all tokens in a response. LEAF keeps the same group-based rollout interface, but replaces sequence-level credit with span-level credit derived from shared prefixes.

Fine-grained and tree-structured credit. Several recent methods seek denser credit than trajectory-level rewards in LLM RL. VinePPO estimates intermediate values by sampling continuations from partial states (Kazemnejad et al., 2025), SPO assigns advantages at the segment level (Guo et al., 2025), and high-entropy analyses suggest that a few uncertain tokens often drive downstream de-

cisions (Wang et al., 2025). Tree-based methods obtain process-level supervision by organizing rollouts into branching structures (LI et al., 2026; Yang et al., 2025; Ji et al., 2026; Hou et al., 2025). LEAF is closest to TreeRL and TEMPO/P2T (Hou et al., 2025; Tran et al., 2025), which estimate prefix-level credit from a fixed group of sampled completions, but it differs in setting, construction, and credit assignment: it performs no additional decoding, targets speech rather than text LLMs, and forks only at a sparse set of high-surprisal locations rather than building the full prefix tree by exact token matching. We formalize in Section 4 why such collision-only matching is problematic.

3 Method

LEAF departs from the dominant execution model in tree-structured RL, which expands a search tree iteratively at inference time. Instead, it exploits structure already present in the batch of i.i.d. traces collected during training: candidate responses that share prefixes are organized retrospectively into a prefix tree, with no additional rollouts. This tree replaces GRPO’s advantage computation. Rather than assigning a single scalar advantage to every token in a response, LEAF assigns piecewise-constant advantages over token spans, localizing credit to where completions diverge (Figure 1).

3.1 Preliminaries: GRPO

Let x be a speech-conditioned input, π_{old} the rollout policy, π_{θ} the optimized policy, and π_{ref} a fixed reference policy. GRPO samples K responses $y^{(1)}, \dots, y^{(K)} \sim \pi_{\text{old}}(\cdot | x)$ and assigns terminal rewards $r_i = r(x, y^{(i)})$. It forms the group-relative advantage $A_i^{\text{GRPO}} = (r_i - \bar{r})/\sigma_r$, where $\bar{r}$ and σ_r are the sample mean and standard deviation, and broadcasts this scalar to every generated token in response i . For token $a_t^{(i)}$ generated from state $s_t^{(i)} = (x, y_{<t}^{(i)})$, define $\rho_{i,t}(\theta) = \pi_{\theta}(a_t^{(i)} | s_t^{(i)})/\pi_{\text{old}}(a_t^{(i)} | s_t^{(i)})$. Given token-level advantages $A_{i,t}$, let $\ell_{i,t}(\theta) = \min\{\rho_{i,t}A_{i,t}, \text{clip}_{\epsilon}(\rho_{i,t})A_{i,t}\}$. The clipped KL-regularized loss is $\mathcal{L}(\theta) = -\mathbb{E}_{i,t}[\ell_{i,t}(\theta)] + \beta \mathbb{E}_{i,t}[\widehat{\text{KL}}_{i,t}]$, where $\widehat{\text{KL}}_{i,t}$ estimates the token-level divergence from π_{ref} and $\text{clip}_{\epsilon}(\rho)$ truncates ρ to the range $[1 - \epsilon, 1 + \epsilon]$. GRPO is recovered by setting $A_{i,t} = A_i^{\text{GRPO}}$ for all t .

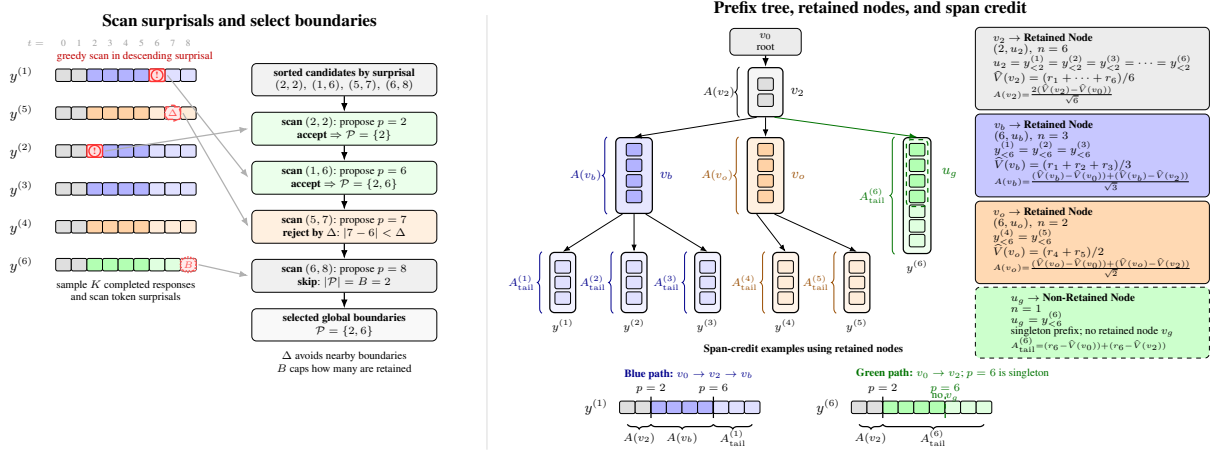


Figure 1: **Overview of LEAF.** From the same K completed responses, LEAF records token surprisals and greedily selects a small set of globally applied boundaries $\mathcal{P}$ using a separation rule. At each selected boundary p , responses are grouped by exact token-prefix equality, forming a tree over the selected boundaries. Non-singleton prefix groups become retained nodes whose values are averages of descendant terminal rewards, while singleton groups are not retained as advantage nodes. The retained nodes on each response path partition the response into spans, and LEAF assigns one advantage to each span.

3.2 Retrospective Tree Credit Assignment

Conceptually, LEAF asks whether shared partial prefixes in the rollout batch predict better or worse final outcomes. It selects at most B token boundaries, groups responses by exact prefix equality at each boundary, retains non-singleton groups as prefix nodes, and assigns advantages to the spans induced by the retained nodes. The only generation cost is the original K responses; B controls the resolution of credit assignment, not the rollout.

LEAF replaces only the advantage estimator $A_{i,t}$. The estimator is based on the process supervision of Hou et al. (2025), which assigns intermediate credit using descendant leaf values, but differs in how the tree is obtained and extends to continuous rewards. The estimator is characterized in Section 4. Rather than branching from intermediate prefixes and decoding new continuations, LEAF samples the same K complete responses used by GRPO and constructs a prefix tree *retrospectively* from shared prefixes within the sampled batch. Thus, the rollout budget remains exactly K responses.

Let response i be $y^{(i)} = (a_0^{(i)}, a_1^{(i)}, \dots, a_{H_i-1}^{(i)})$, where H_i is its length. For a token boundary p , define the length- p prefix $y_{<p}^{(i)} = (a_0^{(i)}, \dots, a_{p-1}^{(i)})$. A selected boundary p is applied to all K responses: responses are grouped together if and only if their token prefixes $y_{<p}^{(i)}$ are exactly identical. Therefore, a selected boundary can produce zero, one, or multiple retained prefix nodes, depending on how many repeated prefix groups appear in the batch. LEAF

replaces the advantage estimation as follows.

Step 1: Responses and surprisals. We sample K complete responses from π_{old} and compute their terminal rewards $r_1, \dots, r_K$. During generation, for each $i \in [K]$, we record the token surprisal $z_{i,t} = -\log \pi_{\text{old}}(a_t^{(i)} \mid x, y_{<t}^{(i)})$, for $0 \leq t < H_i$. Surprisal is used only to decide where to inspect, without affecting the rollout distribution.

Step 2: Fork boundaries. We use high-surprisal token positions to choose prefix boundaries. Let $\ell_{\min} = \min_{i \in [K]} H_i$ and $\Delta = \max\{2, \lfloor \ell_{\min}/(B+1) \rfloor\}$. The shortest length $\ell_{\min}$ restricts candidates to token boundaries that are present in every sampled response, so that the same boundary p can be applied to all K responses when forming exact prefix groups. The separation parameter Δ prevents the fork budget from being spent on several nearby high-surprisal tokens from the same local uncertainty region, encouraging the selected boundaries to cover distinct parts of the response. We form the candidate list $\mathcal{C} = \{(z_{i,t}, i, t) : i \in [K], 1 \leq t < \ell_{\min}\}$, and sort $\mathcal{C}$ in descending order of $z_{i,t}$. We initialize $\mathcal{P} = \emptyset$ and scan this sorted list greedily. When considering a candidate triple $(z_{i,t}, i, t)$, we propose the boundary $p = t$, corresponding to the prefix immediately before the high-surprisal token $a_t^{(i)}$. The boundary p is accepted if either $\mathcal{P} = \emptyset$ or $\min_{p' \in \mathcal{P}} |p - p'| \geq \Delta$. If accepted, we update $\mathcal{P} \leftarrow \mathcal{P} \cup \{p\}$. The scan stops once $|\mathcal{P}| = B$ or the candidate list is exhausted. Finally, we sort the accepted boundaries and write $\mathcal{P} = \{p_1, \dots, p_b\}$

with $p_1 < \dots < p_b$ and $b \leq B$. Accepting a boundary p does not select a single response-specific prefix, as the same boundary p is applied to all K sampled responses, and grouping is performed by exact equality of the token sequences $y_{<p}^{(i)}$.

Step 3: Retained prefix nodes. For each selected boundary $p \in \mathcal{P}$, we compare the length- p token prefixes of all sampled responses. For a token prefix u , define the descendant index set $I(p, u) = \{i \in [K] : y_{<p}^{(i)} = u\}$, with $n(p, u) = |I(p, u)|$. The equality $y_{<p}^{(i)} = u$ is exact token-prefix matching: two responses are grouped together only if their first p tokens are identical. If $n(p, u) \geq 2$, the pair $v = (p, u)$ is retained as a prefix node. Singleton prefixes are discarded because they do not provide a reusable prefix-level value estimate. Let $\mathcal{N}_{\text{fork}}$ denote the set of retained prefix nodes. For $v = (p, u)$, write $p(v) = p$, $u(v) = u$, $I(v) = I(p, u)$, and $n(v) = n(p, u)$. We also define a virtual root node v_0 , corresponding to the empty prefix before any token is generated, with $p(v_0) = 0$, $I(v_0) = [K]$, and $n(v_0) = K$. Its value is the batch-average reward $\hat{V}(v_0) = \bar{r}$. Each retained node v receives the empirical prefix value $\hat{V}(v) = \frac{1}{n(v)} \sum_{i \in I(v)} r_i$. Intuitively, $\hat{V}(v)$ estimates how good completions tend to be after passing through the exact token prefix represented by node v , while $\hat{V}(v_0)$ is the prompt-level average over all sampled completions.

Step 4: Span-level advantages. For each response i , collect all retained nodes whose prefixes lie on the response: $\mathcal{N}_i = \{v = (p, u) \in \mathcal{N}_{\text{fork}} : y_{<p}^{(i)} = u\}$. Sort them by boundary position to obtain the path $v_1^{(i)}, \dots, v_{q_i}^{(i)}$, where $0 < p(v_1^{(i)}) < \dots < p(v_{q_i}^{(i)}) < H_i$, and q_i is the number of retained prefix nodes on response i . Set $v_0^{(i)} = v_0$ and $p(v_0^{(i)}) = 0$. This path partitions response i into spans between consecutive retained prefix nodes and a final tail span. For an internal span ending at node $v_j^{(i)}$, LEAF uses the global-plus-local advantage $A_j^{(i)} = (\text{GA}(v_j^{(i)}) + \text{LA}(v_j^{(i)})) / \sqrt{n(v_j^{(i)})}$, where $\text{GA}(v_j^{(i)}) = \hat{V}(v_j^{(i)}) - \hat{V}(v_0)$ and $\text{LA}(v_j^{(i)}) = \hat{V}(v_j^{(i)}) - \hat{V}(v_{j-1}^{(i)})$. GA measures whether the prefix is better than the prompt-level average, and LA measures whether the current span moved the response to a better prefix than the previous retained prefix on the same path. The factor $1/\sqrt{n(v_j^{(i)})}$ is a multiplicity correction: if many sampled responses share the same prefix, the same

token span appears many times in the batch, and the correction prevents that duplicated span from dominating the update solely because it is repeated.

For each $j = 1, \dots, q_i$, every token $t \in [p(v_{j-1}^{(i)}), p(v_j^{(i)})]$ receives the same advantage $A_j^{(i)}$. Thus, the advantage is not constant over the whole response; it is piecewise constant over spans between retained prefix nodes. For the final tail span, we treat the terminal response as the leaf value. Let $v_{\text{last}}^{(i)} = v_{q_i}^{(i)}$ if $q_i > 0$, and $v_{\text{last}}^{(i)} = v_0$ otherwise. We set $A_{\text{tail}}^{(i)} = (r_i - \hat{V}(v_0)) + (r_i - \hat{V}(v_{\text{last}}^{(i)}))$. Every token $t \in [p(v_{\text{last}}^{(i)}), H_i]$ receives $A_{\text{tail}}^{(i)}$. If response i has no retained prefix node, then the whole response is a single tail span and this gives $2(r_i - \hat{V}(v_0))$, which has the same direction as the GRPO advantage before normalization. In practice, the raw span advantages are normalized across the sampled batch before entering the loss.

4 Theoretical Insights

We provide theoretical insights into three design choices of LEAF: the span-level advantage estimator, the fork budget B , and surprisal-based boundary selection. Specifically, we show that the empirical advantage estimator that LEAF uses preserves the correct update direction, we characterize the tradeoff regarding the value of B and highlight the issues with pure prefix matching. We state concise informal versions of the results here and defer the formal statements and proofs to Appendix E.

Theorem 1 (Validity of span-level advantages). *For any span ending at prefix U_j , the root-relative difference $V(U_j) - V(U_0)$ and the parent-relative difference $V(U_j) - V(U_{j-1})$ are valid scalar advantages. Under fixed-prefix conditional sampling, $\hat{V}(v)$ is an unbiased estimate of $V(v)$ with variance scaling as $1/n(v)$. Thus, shared-prefix averaging denoises the value before span-level credit.*

Theorem 2 (Fork budget tradeoff). *The fork budget B increases the number of selected boundaries at which prefix values may be estimated. With rollout budget K , at most $B \lfloor K/2 \rfloor$ non-root prefix nodes can be retained. Increasing B gives finer credit assignment but can create more low-support value estimates. With $K = 8$ each boundary can retain at most four nodes; using $B = 2$ keeps the worst-case number of non-root prefix values at 8 while still allowing internal span-level credit.*

Theorem 3 (Pure prefix matching is shallow-bi-

ased). *Prefix-collision mass is monotone non-increasing with boundary depth, so a selector that maximizes collisions alone selects from the earliest high-collision plateau. Thus, pure prefix matching can spend the fork budget on shallow boilerplate prefixes. LEAF separates the two roles: surprisal selects where to inspect, while exact prefix matching decides which prefixes at that boundary have enough support to retain.*

5 Experimental Setup

Datasets. We evaluate LEAF on four speech-language benchmarks covering short-form SQA, long-form SQA, and AST. For short-form SQA, we use Part I of LibriSQA (Zhao et al., 2025), an open-ended QA benchmark over LibriSpeech audio (Panayotov et al., 2015). For long-form SQA, we use two LongAudio-derived settings (Ghosh et al., 2025): DailyTalk (Lee et al., 2023), after removing multiple-choice examples, and a Europarl/VoxPopuli split (Koehn, 2005; Wang et al., 2021a), which we refer to as LongAudio. We restrict both long-form settings to audio segments of at most 40 seconds. For AST, we use CoVoST2 English-to-German (Wang et al., 2021b). We use dataset-provided instructions when available and only adapt input formatting to each SALLM; split details are in Appendix B.2.

Baselines and backbones. We compare LEAF with the state-of-the-art (SOTA) GRPO speech-RL baseline of Elmakies et al. (2026). To our knowledge, this method represents the current SOTA on two of our datasets. For the LongAudio and DailyTalk splits, we are not aware of an established task-specific baseline, and hence, we use GRPO as the strongest applicable reference method. The reported results in Elmakies et al. (2026) use full-parameter tuning; our controlled LEAF-vs.-GRPO experiments use LoRA for both methods to isolate the RL objective under the same trainable-parameter budget. When direct comparison to their reported numbers is possible, we include it; otherwise, we use our LoRA GRPO implementation. For rollouts, we match their reported budget with $K = 8$ responses per prompt. LEAF uses the same rollout budget with fork budget $B = 2$; all other optimization and decoding hyperparameters are matched between LEAF and GRPO and reported in Appendix B.1. We evaluate Granite Speech 3.3 2B/8B (Saon et al., 2025), Granite-4.0-1B-Speech (Team, 2026), and Qwen2-Audio-7B

(Chu et al., 2024). Although Qwen is older than the SOTA Granite-4.0, we include it to evaluate whether LEAF generalizes across model families.

Evaluation metrics. To ensure fair comparison with Elmakies et al. (2026), we report BLEU, ROUGE-1/2/L, METEOR, and BERTScore-F1 using temperature 0.9 and top- $p = 0.9$ decoding. To complement automatic metrics, we use M-Prometheus-14B (Pombal et al., 2025) for rubric-grounded Likert-5 scoring and GEMBA-style DA-100 (Kocmi and Federmann, 2023) with Qwen2.5-14B-Instruct (Qwen et al., 2025) for continuous per-item scores. DA-100 scores are averaged over $N = 5$ judgments at $T = 0.7$ and used for tail-risk analysis. We also perform matched A/B comparisons between LEAF and GRPO. For QA tasks, we use JudgeLM-13B (Zhu et al., 2025), while for CoVoST2 we use M-Prometheus-14B. Each pair is judged in both candidate orders, with ties allowed. The pairwise judges are decoded greedily.

Statistical analysis. We report 95% confidence intervals for judge means and CVaR@ α using 1000 non-parametric bootstrap resamples. Tail risk is measured by CVaR@ α , the mean score over the worst α fraction of examples (Rockafellar and Uryasev, 2000). Because Likert-5 CVaR@10 saturates at the minimum score whenever at least 10% of examples receive score 1, we use the continuous DA-100 scores for the main tail-risk analysis.

6 When Does LEAF Work?

Before presenting the main results, we ask whether LEAF has the raw structure it needs to operate. It relies on an empirical property of i.i.d. rollout groups: sampled responses must share exact prefixes often enough, and at sufficiently deep token positions, for the induced virtual forks to provide useful credit-assignment signal. We call this property *forkability*. To define it, consider K rollouts for a fixed prompt. For a selected boundary t , group responses by exact equality of their length- t prefixes. A *fork-fire* occurs at t if at least one prefix group has size at least two. A fork-fire is *usable* if at least one such non-singleton group has non-trivial reward variation, $\max_{i,j \in G} |r_i - r_j| > 0$ for some retained prefix group G . Thus, usable fork-fires correspond to shared prefixes whose descendant responses lead to distinguishable outcomes, rather than merely duplicate completions.

Open-form speech understanding is a natural regime for forkability: conditioned on the same

audio, sampled responses can share substantial prefixes while diverging around localized acoustic or semantic decisions. If those alternative continuations produce different rewards, the resulting rollout structure provides precisely the intermediate signal that LEAF exploits. The converse also clarifies when LEAF should offer little advantage over GRPO. In very short-output tasks, such as direct multiple-choice prediction, responses may diverge before informative internal forks can form, leaving little opportunity for finer credit assignment. Forkability is not inherently specific to speech, however; we view it as a task- and policy-dependent condition that may also arise in other modalities. Establishing when it holds in settings such as long-form LLM reasoning is outside the scope of this work.

We measure forkability across four datasets and three backbones, a 3×4 design with $\approx 78,915$ fork attempts. The usable fork-fire rate is $\geq 72.9\%$ in every one of the twelve cases, while pooled rates differ by only 1.4 pp across backbones. In contrast, rates vary substantially across tasks, suggesting that forkability in our setting is primarily driven by task structure rather than backbone choice.

Direct measurement. We run LEAF from each of the twelve cells and run it on a held-out validation+test split of 3000–3400 examples per cell. Table 1 reports the pooled rates with 95% per-position bootstrap CIs. The pooled-by-backbone rates differ by 1.4 pp., recovering backbone-invariance. The pooled-by-task rates span 19.2 pp and follow task difficulty (easier QA tasks fork more reliably than translation and long-form QA). Across all twelve cells, r_f never drops below 72.9%; the per-cell breakdown (Tab. 10) is in the appendix. Figure 2 plots usable rate vs. token position; for each task, the three backbone curves overlap within their 95% CIs at every position, giving an analogue of the pooled claim.

Forks branch in distinct outcomes, and reach the response. A high rate alone is not enough: fires must produce non-singleton groups with discriminating rewards, and must occur deep enough into the response to discriminate body tokens rather than just the opening. Per-cell statistics in the appendix (Tab. 11, 12) confirm both: mean group sizes lie in 4.0–6.2 of the $K = 8$ budget, within-group BLEU spread is strictly positive in every cell, and the 90%-cumulative depth P_{90} ranges from 14 to 39 token positions across cells. Forks reach into the body, particularly on long-form QA where $P_{90} \geq 35$ across all three backbones.

Table 1: Usable fork-fire rate r_f , pooled by backbone (top) and by task (middle). Confidence intervals are per-position non-parametric bootstrap ($n_{\text{boot}} = 5000$); positions are the unit of resampling because rollouts within an example are correlated through the sampler.

Usable fork-fire rate r_f (% , [95% CI])	
<i>Pooled by backbone (across 4 tasks)</i>	
Granite 3.3-2B	84.6 _[80.6, 88.0]
Granite 3.3-8B	84.6 _[80.8, 87.6]
Granite 4.0-1B	86.0 _[82.4, 88.9]
<i>Pooled by task (across 3 backbones)</i>	
LibriSQA	93.2 _[90.9, 94.7]
DailyTalk	91.6 _[88.1, 94.1]
CoVoST2	81.1 _[75.9, 85.1]
LongAudio	74.0 _[70.1, 77.6]
All 12 cells	85.1_[82.9, 86.9]

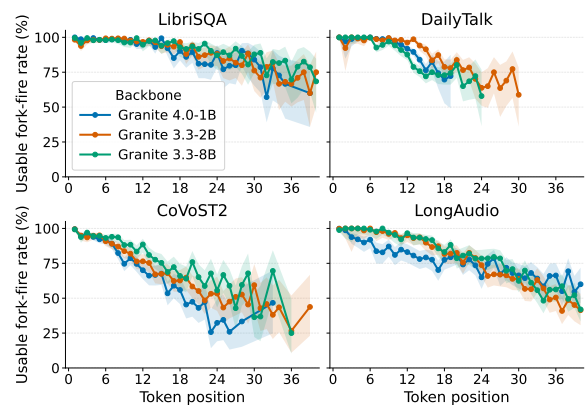


Figure 2: Usable fork-fire rate vs. token position, one panel per task, three backbones overlaid. Shaded bands are pointwise 95% CIs; positions with fewer than 15 attempts are dropped to suppress small- n noise; x -axis is clipped to position 40. Backbones overlap within their CIs in every panel, supporting backbone-invariance.

7 Results

We organize the results around two comparisons. First, we compare LEAF against the published speech-RL state of the art on LibriSQA, where the evaluation setup is closely reproducible. Second, we compare LEAF against our own LoRA GRPO baseline across all datasets, which controls for prompts, decoding, rollout budget, trainable-parameter budget, and evaluation pipeline.

7.1 Comparison Against the State-of-the-art

Table 2 includes the published full-parameter GRPO results of Elmakies et al. (2026) on LibriSQA. Despite using only LoRA adapters, LEAF improves over their full-parameter GRPO numbers on the matched Granite Speech backbones. On Granite 3.3 2B, LEAF improves over full-parameter GRPO while training only

Table 2: End-to-end results across datasets, backbones, and training methods. Each row is the best-val BLEU checkpoint. M-Prometheus Likert is a 1–5 rubric-grounded judge; GEMBA-DA is a 0–100 continuous score.

Dataset	Backbone	Method	BLEU	R-1	R-2	R-L	MTR	BS-F1	Toks	Likert ₁₋₅ [CI]	DA ₀₋₁₀₀ [CI]
LibriSQA	Granite 3 2B	base	28.02	56.80	40.35	51.39	53.02	91.18	22.3	2.65 [2.59, 2.71]	70.52 [69.27, 71.71]
		+Full GRPO	44.90	68.56	53.35	64.88	68.48	94.45	—	—	—
		+LoRA GRPO	40.73	67.25	52.33	63.76	66.78	94.01	19.7	3.20 [3.15, 3.26]	84.89 [84.00, 85.72]
		+LEAF	45.82	68.84	54.19	65.60	68.53	94.64	18.5	3.34 [3.29, 3.39]	86.64 [85.83, 87.38]
	Granite 3 8B	base	24.62	54.33	38.34	48.43	53.29	90.87	26.1	2.28 [2.22, 2.33]	64.57 [63.26, 65.81]
		+Full GRPO	46.40	69.57	57.49	66.16	69.61	94.76	—	—	—
		+LEAF	47.33	69.65	57.69	66.42	69.71	94.80	18.6	3.41 [3.35, 3.46]	87.67 [86.95, 88.45]
		base	27.78	56.43	40.00	51.00	52.97	91.13	22.5	2.62 [2.56, 2.68]	69.99 [68.71, 71.18]
	Granite 4.0 1B	+LEAF	44.61	67.65	52.60	64.11	67.37	94.36	18.6	3.21 [3.15, 3.27]	84.38 [83.44, 85.30]
		base	12.93	38.14	24.99	34.59	40.12	88.19	45.0	2.09 [2.03, 2.15]	55.74 [54.23, 57.25]
		+LoRA GRPO	40.95	66.42	51.59	63.05	66.89	94.06	20.1	3.32 [3.27, 3.38]	83.62 [82.63, 84.48]
		+LEAF	45.85	68.94	54.32	65.59	68.65	94.64	18.8	3.43 [3.37, 3.48]	85.78 [84.94, 86.58]
DailyTalk	Granite 3 2B	base	2.00	15.52	4.79	12.27	22.94	84.04	71.8	1.30 [1.28, 1.33]	44.04 [43.13, 44.96]
		+LoRA GRPO	17.56	46.91	27.97	43.35	43.70	91.63	19.1	2.56 [2.52, 2.60]	62.62 [61.60, 63.66]
		+LEAF	22.61	51.98	31.14	48.14	48.46	92.49	15.3	2.86 [2.82, 2.90]	68.99 [68.11, 69.88]
LongAudio	Granite 3 2B	base	8.46	31.01	13.91	23.67	34.76	86.70	70.1	2.01 [1.94, 2.07]	66.61 [65.21, 67.96]
		+LoRA GRPO	22.44	49.60	31.77	43.56	45.59	90.83	40.5	2.43 [2.37, 2.49]	67.11 [65.79, 68.46]
		+LEAF	31.12	56.24	37.56	50.27	50.62	92.63	30.2	2.93 [2.88, 2.97]	70.21 [69.14, 71.30]
CoVoST2	Granite 3 2B	base	11.45	23.34	12.88	22.16	25.44	79.78	15.5	1.08 [1.07, 1.08]	45.07 [44.69, 45.47]
		+LoRA GRPO	27.77	55.68	33.76	52.59	53.28	85.43	22.1	1.92 [1.90, 1.94]	64.94 [64.47, 65.38]
		+LEAF	30.57	59.20	37.21	56.04	56.14	86.28	21.5	2.13 [2.11, 2.16]	69.09 [68.64, 69.54]
		base	16.66	30.55	18.30	28.93	31.98	81.16	17.1	1.14 [1.14, 1.15]	49.32 [48.94, 49.72]
	Granite 3 8B	+LoRA GRPO	31.95	59.79	38.01	56.48	57.41	86.70	22.3	2.25 [2.23, 2.27]	73.05 [72.64, 73.47]
		+LEAF	33.87	61.58	40.09	58.31	59.11	87.20	21.9	2.42 [2.40, 2.45]	75.01 [74.59, 75.43]
		base	11.80	23.67	13.15	22.51	25.91	79.91	15.5	1.07 [1.07, 1.08]	44.78 [44.42, 45.15]
		+LEAF	31.12	59.04	37.12	55.84	56.74	86.44	21.9	2.14 [2.11, 2.16]	68.39 [67.92, 68.83]

133.7M parameters instead of 3.01B, a $22.5\times$ reduction in trainable parameters. On Granite 3.3 8B, LEAF trains 218.9M parameters instead of 8.65B, a $39.5\times$ reduction, while again improving the reported LibriSQA performance. The smaller Granite-4.0-1B-Speech model trained with LEAF uses only 102.2M trainable parameters and approaches the published full-parameter Granite 3.3 2B GRPO result, further suggesting that improved credit assignment can compensate for a substantially smaller trainable-parameter budget.

We restrict cross-paper comparison to LibriSQA. For CoVoST2, the frozen-base Granite 3.3 2B/8B results reported by Elmakies et al. (2026) are substantially higher than the frozen-base performance obtained under our prompts and input formatting. This mismatch suggests that the CoVoST2 setups differ in prompt template or input formatting, not only in the RL algorithm. Thereupon, we treat our LoRA GRPO implementation as the controlled CoVoST2 baseline and do not include the published CoVoST2 numbers, and we provide all prompts that were used in our experiments in the Appendix.

7.2 Controlled Comparison Against GRPO

Table 2 gives the controlled comparison between LEAF and LoRA GRPO across all datasets and

backbones. LEAF improves over LoRA GRPO in every matched setting, with BLEU gains ranging from +1.92 to +8.68. The largest gains occur on the long-form QA settings, where broadcasting a single terminal-reward advantage across the entire response is most likely to dilute span-level signals.

The judge-based metrics show the same trend. LEAF improves GEMBA-DA in every matched cell, with gains ranging from +1.75 to +6.37, and likewise improves Likert scores throughout. Thus, the gains are not limited to n-gram overlap metrics, but persist under both rubric-grounded and continuous LLM-as-judge evaluation.

Tail-risk. Table 3 shows that LEAF improves worst-case behavior in addition to average quality. Across the matched QA settings, LEAF improves CVaR@10 in every case, with gains ranging from +4.94 to +10.25. On CoVoST2, LEAF likewise improves CVaR@25 and reduces the fraction of examples scoring below 50 for both Granite backbones. These results suggest that LEAF reduces severe failures, consistent with localizing negative credit to harmful spans rather than broadcasting it across the entire response.

Pairwise preference. Table 4 reports direct A/B comparisons between LEAF and GRPO on matched examples. LEAF achieves a positive net

Table 3: Tail-risk analysis on GEMBA-DA-100 continuous scores. CVaR@ α is the mean of the worst α fraction of items; higher is better, except frac< 50 where lower is better. Bootstrap 95% CIs use 1000 resamples.

Dataset	Backbone	Method	N	Mean [95% CI]	CVaR@10 [CI]	CVaR@25 [CI]	frac < 50
LibriSQA	Qwen2-Audio	base	2620	55.74 [54.23, 57.25]	0.00 [0.00, 0.00]	1.25 [0.80, 1.88]	38.59%
		+LoRA GRPO	2620	83.62 [82.63, 84.48]	19.06 [15.82, 23.40]	51.04 [47.74, 54.08]	10.31%
		+LEAF	2620	85.78 [84.94, 86.58]	29.31 [24.82, 33.73]	58.19 [55.46, 61.01]	7.86%
	Granite 3 2B	base	2620	70.52 [69.27, 71.71]	6.25 [5.18, 7.60]	22.12 [19.98, 24.25]	24.77%
		+LoRA GRPO	2620	84.89 [84.00, 85.72]	24.27 [20.76, 28.27]	54.04 [50.99, 56.91]	9.62%
		+LEAF	2620	86.64 [85.83, 87.38]	32.04 [28.03, 36.30]	59.56 [56.81, 62.06]	7.40%
	Granite 3 8B	base	2620	64.57 [63.26, 65.81]	5.15 [4.41, 6.06]	16.96 [15.34, 18.63]	31.56%
		+LEAF	2620	87.67 [86.95, 88.45]	36.04 [31.61, 40.66]	62.78 [60.14, 65.49]	6.45%
	Granite 4.0 1B	base	2620	69.99 [68.71, 71.18]	5.85 [4.80, 7.13]	21.57 [19.42, 23.64]	25.61%
		+LEAF	2620	84.38 [83.44, 85.30]	21.15 [17.60, 25.38]	52.38 [49.39, 55.18]	9.77%
DailyTalk	Granite 3 2B	base	3150	44.04 [43.13, 44.96]	2.05 [1.68, 2.49]	8.23 [7.40, 9.09]	52.70%
		+LoRA GRPO	3150	62.62 [61.60, 63.66]	6.92 [6.09, 7.82]	19.74 [18.31, 21.28]	30.86%
		+LEAF	3150	68.99 [68.11, 69.88]	11.86 [10.38, 13.51]	30.44 [28.49, 32.51]	20.73%
LongAudio	Granite 3 2B	base	1632	66.61 [65.21, 67.96]	1.26 [0.79, 2.13]	20.65 [17.08, 24.31]	21.57%
		+LoRA GRPO	1632	67.11 [65.79, 68.46]	8.89 [7.15, 11.00]	26.88 [24.23, 29.82]	22.61%
		+LEAF	1632	70.21 [69.14, 71.30]	16.62 [13.96, 19.47]	36.31 [33.69, 39.17]	16.91%
CoVoST2	Granite 3 2B	base	15485	45.07 [44.69, 45.47]	5.81 [5.47, 6.14]	13.18 [12.80, 13.58]	57.02%
		+LoRA GRPO	15485	64.94 [64.47, 65.38]	10.65 [10.25, 11.10]	21.53 [20.93, 22.08]	31.35%
		+LEAF	15485	69.09 [68.64, 69.54]	12.27 [11.77, 12.81]	25.34 [24.62, 26.11]	25.98%
	Granite 3 8B	base	15485	49.32 [48.94, 49.72]	6.42 [6.05, 6.81]	14.73 [14.31, 15.17]	49.17%
		+LoRA GRPO	15485	73.05 [72.64, 73.47]	14.97 [14.27, 15.67]	31.64 [30.73, 32.53]	20.89%
		+LEAF	15485	75.01 [74.59, 75.43]	15.08 [13.82, 15.29]	34.02 [32.96, 35.17]	18.66%
	Granite 4.0 1B	base	15485	44.78 [44.42, 45.15]	7.02 [6.67, 7.40]	14.28 [13.95, 14.70]	58.22%
		+LEAF	15485	68.39 [67.92, 68.83]	12.24 [11.72, 12.76]	24.70 [24.04, 25.41]	26.99%

Table 4: Pairwise A/B evaluation of LEAF vs. GRPO with position-swap consistency aggregation. Consistency, LEAF win, GRPO win, and tie are reported as fractions of all evaluated pairs; LEAF win (decided) is computed only among consistent non-tie pairs.

Judge	Dataset	Backbone	N	Consist.	LEAF win	GRPO win	Tie	Net adv.	LEAF win (decided)
JudgeLM	LibriSQA	Granite 3 2B	2620	80.0%	11.4%	3.8%	64.8%	+7.6%	75.1%
		Qwen2-Audio	2620	82.0%	7.8%	3.9%	70.3%	+3.9%	66.6%
	DailyTalk	Granite 3 2B	3150	68.2%	24.0%	12.6%	31.6%	+11.4%	65.5%
	LongAudio	Granite 3 2B	1632	57.9%	30.5%	9.3%	18.1%	+21.2%	76.7%
M-Prom.	CoVoST2	Granite 3 2B	15485	60.7%	39.2%	21.5%	0.0%	+17.7%	64.6%
		Granite 3 8B	15485	54.8%	34.4%	20.4%	0.0%	+14.0%	62.8%

win advantage in every evaluated setting, ranging from +3.9 to +21.2 percentage points across JudgeLM and M-Prometheus evaluations. Thus, the improvements in judge scores are also reflected in direct output-level preferences for LEAF.

Length and runtime efficiency. LEAF produces more compact outputs than GRPO, reducing predicted tokens by 6.1 – 6.5% on LibriSQA, and by 19.9% and 25.4% on DailyTalk and LongAudio. Importantly, LEAF reuses the same sampled completions as GRPO and requires no additional decoding, model forward passes, process reward model, or architectural changes, adding only vectorized batch post-processing over at most B boundaries. In our Granite 2B run, LEAF takes 8.36s/step compared with 12.52s/step for GRPO, a 33.2% reduction consistent with its shorter generations.

8 Ablation Studies

We now examine the design choices underlying LEAF and shed light into our theoretical insights empirically. All ablations use Granite Speech 3.3 2B trained on LibriSQA for a single epoch, with the same hyperparameters as our other experiments. We use a single training seed for these controlled experiments. For the first two ablations, we report BLEU, DA-100, and Likert scores, and summarize performance using a normalized average (Norm. Avg.) computed as $\text{Norm. Avg.} = (\text{BLEU} + \text{DA-100} + 25(\text{Likert} - 1))/3$.

8.1 Effect of the Fork Budget

Regarding the fork budget, our design uses $B = 2$ as a natural operating point. With only one retained boundary along a response path ($B = 1$), the global and local advantages coincide. Thus, a single boundary cannot provide a genuinely dis-

Table 5: Fork-budget ablation on LibriSQA with Granite Speech 3.3 2B with one training epoch. The trend in performance is non-monotone; $B = 2$ performs best.

B	BLEU	DA-100	Likert	Norm. Avg.
2	43.91	85.47	3.23	61.71
4	39.37	72.73	2.84	52.70
6	40.45	78.69	3.00	56.38
8	41.22	81.12	3.07	58.03

Table 6: Advantage-form ablation on LibriSQA with Granite Speech 3.3 2B with one training epoch. Local and global advantage are both valid forms with similar performance, while their sum performs best.

Advantage	BLEU	DA-100	Likert	Norm. Avg.
GA	43.24	84.46	3.19	60.82
LA	43.32	84.65	3.20	60.99
GA+LA	43.91	85.47	3.23	61.71

tinct parent-relative signal in addition to the root-relative signal. A second retained boundary is the smallest setting in which the two can differ, making $B = 2$ the minimum fork budget needed to realize LEAF’s credit decomposition. At the same time, Theorem 2 highlights a trade-off: increasing B improves credit-assignment resolution but can introduce more low-support prefix-value estimates. We therefore examine whether increasing $B \geq 2$ is beneficial.

Table 5 shows that $B = 2$ performs best across all three metrics and obtains the highest normalized average. Increasing B does not monotonically improve performance: although additional boundaries permit a finer decomposition of the response, all larger budgets remain below $B = 2$. This behavior is consistent with the support–resolution tradeoff in Theorem 2.

8.2 Effect of the Advantage Form

LEAF combines two complementary prefix-level signals, GA and LA. Theorem 1 establishes that both are individually valid span-level credit signals. Hence, we ablate the estimator by retaining only GA, only LA, or their combination. As shown in Table 6, GA and LA achieve very similar performance when used independently, consistent with Theorem 1. Their combination performs best across all metrics, reflecting complementary information: GA evaluates a prefix relative to the rollout group, while LA captures its incremental change from the preceding prefix.

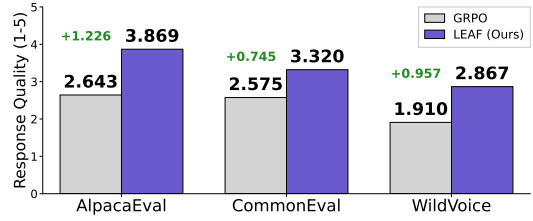


Figure 3: Cross-benchmark evaluation on three open-ended VoiceBench subsets using M-Prometheus-14B as the judge. Both checkpoints are trained only on LibriSQA with Granite Speech 3.3 2B for one epoch. LEAF outperforms GRPO in all cases.

8.3 Cross-Benchmark Generalization

We examine whether the improvements obtained by LEAF are specific to the LibriSQA data used for training. We evaluate the same one-epoch Granite Speech 3.3 2B GRPO and LEAF checkpoints, on the AlpacaEval, CommonEval, and WildVoice subsets of VoiceBench (Chen et al., 2026) and we use M-Prometheus-14B, due to its competitive judging performance (Pombal et al., 2025), to assign 1–5 quality scores to the generated responses. Figure 3 shows that LEAF improves over GRPO on all three unseen benchmarks, and the average score increases from 2.373 to 3.352. These results therefore indicate that LEAF’s gains transfer to unseen open-ended speech settings, demonstrating that its finer span-level credit assignment improves more general speech-conditioned response behavior.

9 Conclusion

In this work, we introduced LEAF, a retrospective tree-based RL method for SALLM post-training. LEAF reuses the same i.i.d. rollout groups collected by GRPO, detects shared prefixes at high-surprisal boundaries, and converts descendant rewards into span-level advantages. This yields process-aware credit assignment without extra decoding, online tree search, or a separate process reward model. Across speech QA and translation tasks, LEAF consistently improves over LoRA GRPO under the same rollout and trainable-parameter budget, while also improving tail behavior, pairwise preferences, and length and runtime efficiency. Importantly, on LibriSQA, LEAF further improves over SOTA full-parameter GRPO while training only LoRA adapters. Future work includes richer reward models, multilingual speech tasks, and adaptive boundary selection.

Limitations

We note several limitations. First, LEAF’s effectiveness is regime-dependent: its mechanism requires that i.i.d. rollouts share prefixes often enough to populate non-trivial fork groups, and the absolute fork rate depends on the policy. We therefore state our claims comparatively (backbone-invariance, task-monotonicity) and as a lower bound ($r_f \geq 72.9\%$ in every cell) rather than as an exact rate. Second, our reward signal is BLEU score, following prior speech-RL work; smoother rewards may yield further gains we do not measure. Third, hardware constraints limited us to three NVIDIA A40 GPUs (48 GB) with a single training seed per (model, method) and thus do not characterize training-seed variance. Moreover, our ablations also used a single seed, and therefore interpret them primarily as evidence for the qualitative behavior of the proposed design choices. Finally, all speech inputs are English, so our findings may not transfer to non-English source speech or other modalities.

Potential Risks and Societal Impact

LEAF is a post-training method rather than a deployed speech system, but improving speech understanding can amplify risks associated with downstream SALLM use. First, model errors can misrepresent the content or intent of source audio and propagate into user-facing decisions; our benchmark improvements should therefore not be interpreted as establishing reliability in high-stakes settings. Second, we evaluate only English source speech and do not study performance across accents, dialects, demographic groups, or non-English languages. Consequently, downstream systems may exhibit uneven error rates for under-represented speakers. Third, stronger speech understanding is inherently dual-use: large-scale analysis of recorded speech could facilitate unwanted monitoring or surveillance and may expose sensitive information contained in speech. Finally, multi-modal speech training carries non-trivial compute and energy costs. LEAF does not require additional rollouts or online tree expansion beyond the fixed GRPO rollout batch, limiting its incremental generation cost relative to GRPO, but training and evaluating the underlying SALLMs remains computationally expensive. Deployment-oriented work should therefore evaluate privacy, demographic robustness, and application-specific safety before use in consequential settings.

AI Usage

AI-based assistants were used for limited editing, wording suggestions, and debugging assistance during experiment development and manuscript preparation. All technical content, experimental design, analysis, and conclusions were verified and finalized by the authors.

Acknowledgements

AG gratefully acknowledges Moulik Choraria for introducing him to tree-based methods for post-training large language models. This work was supported in part by a grant from the Institute for Information & Communications Technology Planning & Evaluation (IITP) to KAIST and in part by the National Science Foundation under grant #2106727. The authors also acknowledge the Advanced Cyberinfrastructure Coordination Ecosystem: Services & Support (ACCESS) program (Boerner et al., 2023) for providing computing allocations and support through allocation CIS260224 (NSF grants #2138259, #2138286, #2138307, #2137603, and #2138296).

References

- Siddhant Arora, Kai-Wei Chang, Chung-Ming Chien, Yifan Peng, Haibin Wu, Yossi Adi, Emmanuel Dupoux, Hung yi Lee, Karen Livescu, and Shinji Watanabe. 2025. [On the landscape of spoken language models: A comprehensive survey](#). *Transactions on Machine Learning Research*.
- Timothy J Boerner, Stephen Deems, Thomas R Furlani, Shelley L Knuth, and John Towns. 2023. Access: Advancing innovation: Nsf’s advanced cyberinfrastructure coordination ecosystem: Services & support. In *Practice and experience in advanced research computing 2023: Computing for the common good*, pages 173–176.
- Yiming Chen, Xianghu Yue, Chen Zhang, Xiaoxue Gao, Robby T. Tan, and Haizhou Li. 2026. [Voicebench: Benchmarking llm-based voice assistants](#). *Transactions of the Association for Computational Linguistics*, 14:378–398.
- Yunfei Chu, Jin Xu, Qian Yang, Haojie Wei, Xipin Wei, Zhifang Guo, Yichong Leng, Yuanjun Lv, Jinzheng He, Junyang Lin, Chang Zhou, and Jingren Zhou. 2024. [Qwen2-audio technical report](#). *Preprint*, arXiv:2407.10759.
- Avishai Elmakies, Hagai Aronowitz, Nimrod Shabtay, Eli Schwartz, Ron Hoory, and Avihu Dekel. 2026. [Advancing speech understanding in speech-aware language models with grpo](#). In *ICASSP 2026*

- 2026 *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 17797–17801.
- Sreyan Ghosh, Arushi Goel, Jaehyeon Kim, Sonal Kumar, Zhifeng Kong, Sang-gil Lee, Chao-Han Yang, Ramani Duraiswami, Dinesh Manocha, Rafael Valle, and Bryan Catanzaro. 2025. [Audio flamingo 3: Advancing audio intelligence with fully open large audio language models](#). In *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pages 41819–41886. Curran Associates, Inc.
- Yiran Guo, Lijie Xu, Jie Liu, DAN YE, and Shuang Qiu. 2025. [Segment policy optimization: Effective segment-level credit assignment in RL for large language models](#). In *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pages 114399–114431. Curran Associates, Inc.
- Zhenyu Hou, Ziniu Hu, Yujiang Li, Rui Lu, Jie Tang, and Yuxiao Dong. 2025. [TreeRL: LLM reinforcement learning with on-policy tree search](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12355–12369, Vienna, Austria. Association for Computational Linguistics.
- Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. [LoRA: Low-rank adaptation of large language models](#). In *International Conference on Learning Representations*.
- Yuxiang Ji, Ziyu Ma, Yong Wang, Guanhua Chen, Xiangxiang Chu, and Liaoni Wu. 2026. [Tree search for LLM agent reinforcement learning](#). In *International Conference on Learning Representations*, volume 2026, pages 87362–87388.
- Amirhossein Kazemnejad, Milad Aghajohari, Eva Portelance, Alessandro Sordani, Siva Reddy, Aaron Courville, and Nicolas Le Roux. 2025. [VinePPO: Refining credit assignment in RL training of LLMs](#). In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 29557–29590. PMLR.
- Tom Kocmi and Christian Federmann. 2023. [Large language models are state-of-the-art evaluators of translation quality](#). In *Proceedings of the 24th Annual Conference of the European Association for Machine Translation*, pages 193–203, Tampere, Finland. European Association for Machine Translation.
- Philipp Koehn. 2005. [Europarl: A parallel corpus for statistical machine translation](#). In *Proceedings of Machine Translation Summit X: Papers*, pages 79–86, Phuket, Thailand.
- Keon Lee, Kyumin Park, and Daeyoung Kim. 2023. [Dailytalk: Spoken dialogue dataset for conversational text-to-speech](#). In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5.
- Yizhi LI, Qingshui Gu, Zhoufutu Wen, Ziniu Li, Ruibin Yuan, Tianshun Xing, Shuyue Guo, Tuney Zheng, Xin Zhou, Xingwei Qu, Wangchunshu Zhou, Zheng Zhang, Wei Shen, Wei Xue, Qian Liu, Chenghua Lin, Jian Yang, Ge Zhang, and Wenhao Huang. 2026. [TreePO: Enhancing policy efficacy and inference efficiency with tree modeling](#). In *Forty-third International Conference on Machine Learning*.
- Chang Liu, Ya-Jun Hu, Ying-Ying Gao, Shi-Lei Zhang, and Zhen-Hua Ling. 2026. [Group relative policy optimization for text-to-speech with large language models](#). In *ICASSP 2026 - 2026 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 16132–16136.
- Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. [Librispeech: An asr corpus based on public domain audio books](#). In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210.
- José Pombal, Dongkeun Yoon, Patrick Fernandes, Ian Wu, Seungone Kim, Ricardo Rei, Graham Neubig, and Andre Martins. 2025. [M-prometheus: A suite of open multilingual LLM judges](#). In *Second Conference on Language Modeling*.
- Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, and 25 others. 2025. [Qwen2.5 technical report](#). *Preprint*, arXiv:2412.15115.
- R. Tyrrell Rockafellar and Stanislav Uryasev. 2000. Optimization of conditional value-at-risk. *Journal of Risk*, 2(3):21–41.
- George Saon, Avihu Dekel, Alexander Brooks, Tohru Nagano, Abraham Daniels, Aharon Satt, Ashish Mittal, Brian Kingsbury, David Haws, Edmilson Morais, Gakuto Kurata, Hagai Aronowitz, Ibrahim Ibrahim, Jeff Kuo, Kate Soule, Luis Lastras, Masayuki Suzuki, Ron Hoory, Samuel Thomas, and 5 others. 2025. [Granite-speech: open-source speech-aware llms with strong english asr capabilities](#). In *2025 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 1–7.
- John Schulman and Thinking Machines Lab. 2025. [Lora without regret](#). *Thinking Machines Lab: Connectionism*. <https://thinkingmachines.ai/blog/lora/>.
- Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. [Deepseekmath: Pushing the limits of mathematical reasoning in open language models](#). *Preprint*, arXiv:2402.03300.
- Prashanth Gurunath Shivakumar, Yile Gu, Ankur Gandhe, and Ivan Buluko. 2025. [Group relative policy optimization for speech recognition](#). In *2025 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 1–7.

IBM Granite Speech Team. 2026. [Granite 4.0 speech](#).

Hieu Tran, Zonghai Yao, and Hong Yu. 2025. [Exploiting tree structure for credit assignment in rl training of llms](#). *Preprint*, arXiv:2509.18314.

Changhan Wang, Morgane Riviere, Ann Lee, Anne Wu, Chaitanya Talnikar, Daniel Haziza, Mary Williamson, Juan Pino, and Emmanuel Dupoux. 2021a. [VoxPopuli: A large-scale multilingual speech corpus for representation learning, semi-supervised learning and interpretation](#). In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 993–1003, Online. Association for Computational Linguistics.

Changhan Wang, Anne Wu, Jiatao Gu, and Juan Pino. 2021b. [CoVoST 2 and Massively Multilingual Speech Translation](#). In *Interspeech 2021*, pages 2247–2251.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xiong-Hui Chen, Jianxin Yang, Zhenru Zhang, Yuqiong Liu, An Yang, Andrew Zhao, Yang Yue, Shiji Song, Bowen Yu, Gao Huang, and Junyang Lin. 2025. [Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning](#). In *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pages 115452–115486. Curran Associates, Inc.

Zhicheng Yang, Zhijiang Guo, Yinya Huang, Xiaodan Liang, Yiwei Wang, and Jing Tang. 2025. [Treerpo: Tree relative policy optimization](#). *Preprint*, arXiv:2506.05183.

Qiyang Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gao-hong Liu, juncai liu, LingJun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, and 17 others. 2025. [Dapo: An open-source llm reinforcement learning system at scale](#). In *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pages 113222–113244. Curran Associates, Inc.

Zihan Zhao, Yiyang Jiang, Heyang Liu, Yu Wang, and Yanfeng Wang. 2025. [Librisqa: A novel dataset and framework for spoken question answering with large language models](#). *IEEE Transactions on Artificial Intelligence*, 6(11):2884–2895.

Lianghui Zhu, Xinggang Wang, and Xinlong Wang. 2025. [JudgeLM: Fine-tuned large language models are scalable judges](#). In *International Conference on Learning Representations*, volume 2025, pages 51257–51296.

A Hardware details

All experiments were run using three NVIDIA A40 GPUs (48 GB).

B Experimental Details

B.1 Training Parameters

Table 7: Shared RL training hyperparameters. Unless otherwise stated, the same settings are used for LEAF and GRPO.

Hyperparameter	Value
LoRA rank / alpha / dropout	64 / 128 / 0.05
Optimizer	Adam
Learning rate	5×10^{-6}
KL coefficient	0.02
Rollout budget K	8 responses per prompt
LEAF fork budget B	2
Rollout batch size	6
Training batch size	6
Micro-batch size	1
Max prompt length	256 tokens
Max response length	200 tokens
Sampling temperature	1.0
Top- p	0.9
Reward	Sentence BLEU

B.2 Dataset Specifics

To create the LibriSQA and the CoVoST2 datasets we follow the process shared in the original works (Wang et al., 2021b; Zhao et al., 2025). In order to create the DailyTalk dataset, we use the prompts and data given in LongAudio (Ghosh et al., 2025) and integrate it accordingly with the original data (Lee et al., 2023), while retaining only examples with audio durations of at most 40 seconds.

Table 8: Dataset splits used in this work.

Dataset	Train	Validation	Test
LibriSQA	85,045	21,196	2,620
DailyTalk	21,891	2,806	3,150
LongAudio	13,374	1,743	1,632
CoVoST2	288,069	15,471	15,485

B.3 Prompt templates

We list the prompt templates used for each dataset. The literal placeholder `<|audio|>` is rewritten at tokenization time to the backbone-specific audio token: a single audio token for Granite Speech and `<|audio_bos|><|AUDIO|><|audio_eos|>` for Qwen2-Audio.

LibriSQA. We use the dataset-provided question field and wrap it with one of the following templates:

- Listen to the audio `<|audio|>` and answer the following question: {question}

Table 9: Number of RL training epochs. All other RL hyperparameters are shared as in Table 7.

Model family	Dataset setting	Epochs
Granite Speech 3.3 2B / 8B	CoVoST2, LibriSQA	2
Granite Speech 3.3 2B	DailyTalk, LongAudio	6
Granite-4.0-1B-Speech	LibriSQA	2
Granite-4.0-1B-Speech	CoVoST2	1
Qwen2-Audio	All evaluated datasets	1

- `<|audio|>` Please answer this question about the spoken passage: {question}
- You will hear speech in `<|audio|>`. Answer: {question}

CoVoST2 English-to-German. We use translation-instruction templates such as:

- `<|audio|>` Listen to the speech and translate it into German.
- Translate the audio `<|audio|>` from English to German.
- You will hear English speech in `<|audio|>`. Provide the German translation.

DailyTalk and LongAudio. For the LongAudio-derived datasets, prompts are inherited from the dataset preparation pipeline. They follow two structural variants:

- `<|audio|>\n{question}`
- `{question}\n<|audio|>`

where {question} is the dataset-provided open-ended question.

B.4 LLM-as-judge protocols

Prediction staging. For each dataset, backbone, method, and checkpoint, we generate a single stochastic prediction file using temperature 0.9, top- $p = 0.9$, and maximum generation length 200. All judge protocols consume the same staged prediction files, so judges are never given access to additional samples.

Grounded Likert-5. We use UNBABEL/M-PROMETHEUS-14B as a rubric-grounded judge on a 1–5 Likert scale. The judge receives the question, reference answer, candidate prediction, and a task-specific rubric. For QA tasks, the rubric emphasizes correctness, groundedness, completeness, and hallucination. For CoVoST2, the rubric emphasizes accuracy, fluency, and style. Decoding is greedy with temperature 0.

GEMBA-DA 0–100. For continuous per-item scoring, we use a GEMBA-style direct-assessment prompt with QWEN2.5-14B-INSTRUCT. Each item is judged $N = 5$ times at temperature 0.7, and we use the per-item mean score as the canonical DA-100 value.

Pairwise judging. For pairwise LEAF-vs.-GRPO comparisons, we use BAAI/JUDGELM-13B-v1.0 for QA tasks and M-Prometheus relative grading for CoVoST2. Each pair is evaluated in both candidate orders. A verdict is counted only when the two swapped evaluations agree; otherwise, the pair is marked inconsistent and excluded from the decided-pair win rate.

C Forkability: per-cell measurements

This appendix expands Section 6 with (i) the per-cell breakdown of the usable fork-fire rate, (ii) the auxiliary fork-group statistics that justify treating fires as discriminating branches rather than near-clones, (iii) the cumulative-position depth quantiles, and (iv) a backbone-organised view of the per-position curves.

C.1 Per-cell usable fork-fire rate

Table 10 gives the 12-cell breakdown that underlies the pooled rows of Tab. 1. The minimum cell rate is 72.9% (Granite-3.3-8B on LongAudio) and the maximum is 94.6% (Granite-4.0-1B on LibriSQA). Within every task the three backbone CIs overlap; between tasks, the LibriSQA / DailyTalk / CoVoST2 / LongAudio ordering recovers the task-monotonicity claim.

Table 10: Per-cell usable fork-fire rate r_f with 95% per-position bootstrap CIs ($n_{\text{boot}} = 5000$). N is the number of validation+test examples measured per cell.

Backbone	Task	N	Selected	Usable	r_f (%; 95% CI)
G. 3.3-2B	LibriSQA	3400	6800	6222	91.5 (86.9–94.1)
G. 3.3-2B	CoVoST2	3400	6799	5364	78.9 (70.5–85.4)
G. 3.3-2B	DailyTalk	3400	6800	6364	93.6 (86.9–96.6)
G. 3.3-2B	LongAudio	3000	6000	4395	73.3 (66.3–79.3)
G. 3.3-8B	LibriSQA	3400	6799	6352	93.4 (89.4–95.8)
G. 3.3-8B	CoVoST2	3000	6000	5055	84.3 (76.1–89.8)
G. 3.3-8B	DailyTalk	3000	6000	5273	87.9 (81.4–93.1)
G. 3.3-8B	LongAudio	3332	6664	4857	72.9 (66.3–78.6)
G. 4.0-1B	LibriSQA	3400	6799	6429	94.6 (90.9–96.6)
G. 4.0-1B	CoVoST2	3400	6790	5472	80.6 (69.7–87.3)
G. 4.0-1B	DailyTalk	3400	6800	6316	92.9 (87.8–96.1)
G. 4.0-1B	LongAudio	3332	6664	5042	75.7 (68.8–81.7)

C.2 Fork-group statistics

Table 11 reports the mean number of rollouts sharing each fired prefix (“Group size”, maximum $K = 8$), the mean within-group BLEU spread (“Reward spread”), the usable-count weighted mean

position of fired forks (“MU pos”), and the number of distinct token positions hosting at least one usable fork (“# pos”). Group sizes range 4.0–6.2 across cells and reward spread is strictly positive in every cell (≥ 0.027), confirming that fired forks discriminate higher- from lower-reward branches rather than collapsing to duplicates.

Table 11: Auxiliary per-cell fork-group statistics.

Backbone	Task	Group size	Reward spread	MU pos	# pos
G. 3.3-2B	LibriSQA	5.40	0.062	12.4	99
G. 3.3-2B	CoVoST2	4.05	0.044	11.1	97
G. 3.3-2B	DailyTalk	6.18	0.030	9.1	70
G. 3.3-2B	LongAudio	4.31	0.036	18.5	78
G. 3.3-8B	LibriSQA	5.75	0.053	13.5	99
G. 3.3-8B	CoVoST2	4.19	0.050	10.0	93
G. 3.3-8B	DailyTalk	4.77	0.040	9.2	35
G. 3.3-8B	LongAudio	4.25	0.034	20.2	78
G. 4.0-1B	LibriSQA	5.48	0.059	9.0	78
G. 4.0-1B	CoVoST2	4.73	0.057	6.9	72
G. 4.0-1B	DailyTalk	5.29	0.027	8.2	53
G. 4.0-1B	LongAudio	3.96	0.030	17.0	77

C.3 Depth profile of usable fork-fires

Table 12 reports cumulative-position quantiles: P_q is the token position by which $q\%$ of the cell’s usable fork-fires have already occurred. The depth profile clusters tightly across backbones within a task, and hence, the forkable window is governed by the task’s input-output coupling, not by the model. Short-output QA tasks concentrate fork-fires within the first ≈ 18 tokens (DailyTalk: $P_{90} \leq 18$ across all three backbones), while long-form LongAudio maintains fork activity past position 35 ($P_{90} = 35$ –39, $P_{99} = 55$ –59).

Table 12: Cumulative-position quantiles of usable fork-fires per cell. The depth profile is essentially constant across backbones within a task.

Backbone	Task	P_{50}	P_{90}	P_{99}	Last	# pos
G. 3.3-2B	LibriSQA	8	28	84	99	99
G. 3.3-2B	CoVoST2	7	22	83	99	97
G. 3.3-2B	DailyTalk	8	18	32	99	70
G. 3.3-2B	LongAudio	17	35	55	99	78
G. 3.3-8B	LibriSQA	7	33	90	99	99
G. 3.3-8B	CoVoST2	6	21	76	99	93
G. 3.3-8B	DailyTalk	9	17	23	53	35
G. 3.3-8B	LongAudio	18	39	59	99	78
G. 4.0-1B	LibriSQA	6	20	41	99	78
G. 4.0-1B	CoVoST2	5	14	38	99	72
G. 4.0-1B	DailyTalk	8	14	18	94	53
G. 4.0-1B	LongAudio	15	37	57	97	77

C.4 Per-backbone view of the per-position curves

Figure 4 replots Fig. 2 with the three backbones placed in separate panels and the four tasks overlaid in each, making the between-task separation visually direct. The four task curves order LibriSQA >

DailyTalk > CoVoST2 > LongAudio at every position beyond the startup region, on every backbone, and the visual analogue of the pooled-by-task ordering is in Tab. 1.

D Additional Experimental Results

D.1 Length Experiments

Table 13: Length efficiency: mean GPT-2 BPE token counts of the prediction (post-hoc, universal tokenizer; same for every row).

Dataset	Backbone	base	GRPO	LEAF
CoVoST2	Granite 3 2B	15.5	22.1	21.5
CoVoST2	Granite 3 8B	17.1	22.3	21.9
DailyTalk	Granite 3 2B	71.8	19.1	15.3
LibriSQA	Granite 3 2B	22.3	19.7	18.5
LibriSQA	Qwen2-Audio	45.0	20.1	18.8
LongAudio	Granite 3 2B	70.1	40.5	30.2

E Theoretical Proofs

This appendix gives the formal statements and proofs for the theoretical claims in the main text. Throughout, we fix a prompt x . Let $Y \sim \pi_\theta(\cdot | x)$ be a random completion and let $R = r(x, Y)$ be its terminal reward. The observed rewards r_i are realizations of R for sampled completions $y^{(i)}$. We assume $R \in [0, R_{\max}]$ when using concentration inequalities.

Assumption 1 (Fixed-prefix conditional sampling). *Fix a boundary p and a prefix $u \in \mathcal{V}^p$. Conditional on $n(p, u) = m$, the rewards of the sampled descendants $\{r_i : y_{<p}^{(i)} = u\}$ are conditionally independent samples from the distribution of R given $Y_{<p} = u$. We write $V(u) = \mathbb{E}[R | Y_{<p} = u]$.*

Assumption 2 (Selected-node concentration). *For each retained node v , let $V_{\text{sel}}(v)$ be its selection-conditional value. We assume that for all $\epsilon > 0$,*

$$\mathbb{P}\left(|\widehat{V}(v) - V_{\text{sel}}(v)| \geq \epsilon\right) \leq 2 \exp\left(-\frac{2n(v)\epsilon^2}{R_{\max}^2}\right).$$

Remark 1 (Ordinary and selection-conditional prefix values). *Assumption 1 is the clean condition for a fixed prefix. Since LEAF selects nodes retrospectively, a selected node can also carry information from the selection event. Assumption 2 therefore states concentration around $V_{\text{sel}}(v)$, the value induced by the selected node. If selection does not distort the descendant reward law beyond conditioning on the retained prefix u , then $V_{\text{sel}}(v) = V(u)$.*

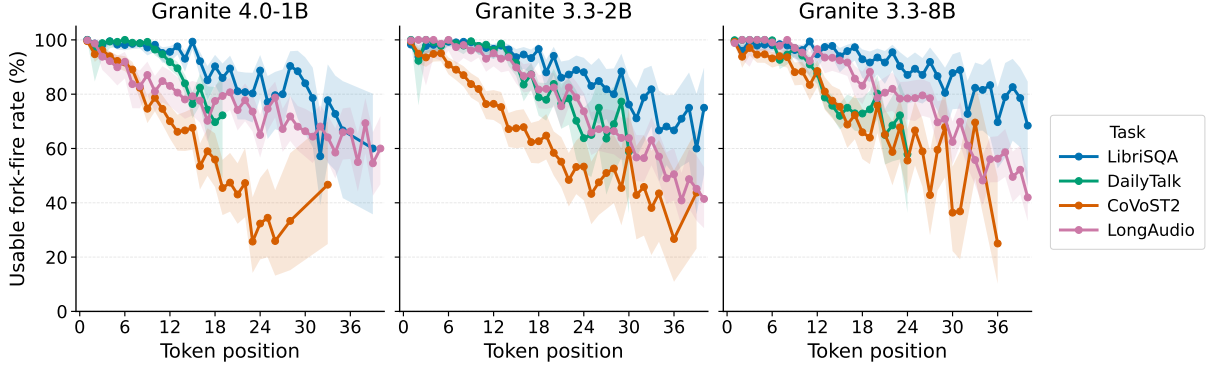


Figure 4: Per-position usable fork-fire rate, one panel per backbone, four tasks overlaid; shaded bands are pointwise 95% CIs.

E.1 Proof of the span-advantage claim

Theorem 4 (Validity of LEAF span advantages). *For a fixed prompt x , let $Y \sim \pi_\theta(\cdot | x)$, $R = r(x, Y)$, and $V(u) = \mathbb{E}[R | Y_{<|u|} = u]$. Consider a path of prefixes $U_0 \prec U_1 \prec \dots \prec U_q$, with $U_0 = \emptyset$, and let $Z_j = \sum_{t=\tau_{j-1}}^{\tau_j-1} \nabla_\theta \log \pi_\theta(A_t | S_t)$ be the score of span j . Define $\text{GA}(U_j) = V(U_j) - V(U_0)$ and $\text{LA}(U_j) = V(U_j) - V(U_{j-1})$. Then $\mathbb{E}[\text{GA}(U_j)Z_j] = \mathbb{E}[\text{LA}(U_j)Z_j] = \mathbb{E}[RZ_j]$. Moreover, if a retained node $v = (p, u)$ has $n(v) = m$ descendants, then under Assumption 1, $\mathbb{E}[\hat{V}(v) | n(v) = m] = V(u)$ and $\text{Var}(\hat{V}(v) | n(v) = m) = \text{Var}(R | Y_{<p} = u)/m$.*

Proof. Fix a span j and a token $t \in [\tau_{j-1}, \tau_j]$. Since $U_j = Y_{<\tau_j}$ contains A_t and its prefix state S_t , the score term is measurable with respect to U_j . By the tower property,

$$\begin{aligned} \mathbb{E}[R \nabla_\theta \log \pi_\theta(A_t | S_t)] \\ = \mathbb{E}[V(U_j) \nabla_\theta \log \pi_\theta(A_t | S_t)]. \end{aligned} \quad (1)$$

For any random variable B that is measurable with respect to S_t ,

$$\begin{aligned} \mathbb{E}[B \nabla_\theta \log \pi_\theta(A_t | S_t)] \\ = \mathbb{E}[B \mathbb{E}[\nabla_\theta \log \pi_\theta(A_t | S_t) | S_t]] = 0, \end{aligned} \quad (2)$$

because $\mathbb{E}[\nabla_\theta \log \pi_\theta(A_t | S_t) | S_t] = 0$. Taking $B = V(U_0)$ in (2) and combining with (1) gives the identity for $\text{GA}(U_j)$. Taking $B = V(U_{j-1})$, which is measurable with respect to S_t for every $t \in [\tau_{j-1}, \tau_j]$, gives the identity for $\text{LA}(U_j)$. Summing over $t = \tau_{j-1}, \dots, \tau_j - 1$ proves $\mathbb{E}[\text{GA}(U_j)Z_j] = \mathbb{E}[\text{LA}(U_j)Z_j] = \mathbb{E}[RZ_j]$.

For the empirical claim, condition on $n(v) = m$ for a fixed prefix node $v = (p, u)$. By Assumption 1, the descendant rewards are conditionally

independent samples from $R | Y_{<p} = u$. Their mean is $V(u)$ and their variance is $\text{Var}(R | Y_{<p} = u)$. Since $\hat{V}(v) = m^{-1} \sum_{i \in I(v)} r_i$, the conditional mean is $V(u)$ and the conditional variance is $\text{Var}(R | Y_{<p} = u)/m$. $\square$

E.2 Proof of the fork-budget claim

Theorem 5 (Fork-budget support-resolution trade-off). *Fix rollout budget K , fork budget $B \geq 1$, and retention threshold $m_0 \geq 2$, with $K \geq m_0$. Let $\mathcal{N}_{\text{fork}}^{(m_0)}$ be the retained non-root prefix nodes obtained by applying at most B selected boundaries to the K sampled responses and keeping only prefix groups with at least m_0 descendants. Then $|\mathcal{N}_{\text{fork}}^{(m_0)}| \leq B \lfloor K/m_0 \rfloor$. For a selected retained node $v = (p, u)$, define its selection-conditional value as $V_{\text{sel}}(v) = \mathbb{E}[R_J | v \in \mathcal{N}_{\text{fork}}^{(m_0)}]$, where J is drawn uniformly from the descendant set $I(v)$. Under Assumption 2, if $n_{\min} = \min_{v \in \mathcal{N}_{\text{fork}}^{(m_0)}} n(v)$, then with probability at least $1 - \delta$, $\max_{v \in \mathcal{N}_{\text{fork}}^{(m_0)}} |\hat{V}(v) - V_{\text{sel}}(v)| \leq R_{\max} \sqrt{\log(2B \lfloor K/m_0 \rfloor / \delta) / (2n_{\min})}$. If selection does not distort the descendant reward law beyond conditioning on the retained prefix, then $V_{\text{sel}}(v) = V(u)$.*

Proof. Fix a selected boundary p . Exact prefix matching partitions the K sampled responses into disjoint groups. Since each retained group has size at least m_0 , there are at most $\lfloor K/m_0 \rfloor$ retained groups at boundary p . Since at most B boundaries are selected,

$$|\mathcal{N}_{\text{fork}}^{(m_0)}| \leq B \left\lfloor \frac{K}{m_0} \right\rfloor.$$

If $\mathcal{N}_{\text{fork}}^{(m_0)} = \emptyset$, the high-probability statement is vacuous. Otherwise, let $n_{\min} = \min_v n(v)$ over

retained nodes. By Assumption 2, for any retained node v ,

$$\mathbb{P}\left(|\widehat{V}(v) - V_{\text{sel}}(v)| \geq \epsilon\right) \leq 2 \exp\left(-\frac{2n(v)\epsilon^2}{R_{\max}^2}\right).$$

Since $n(v) \geq n_{\min}$, a union bound gives

$$\begin{aligned} \mathbb{P}\left(\max_v |\widehat{V}(v) - V_{\text{sel}}(v)| \geq \epsilon\right) &\leq 2|\mathcal{N}_{\text{fork}}^{(m_0)}| \exp\left(-\frac{2n_{\min}\epsilon^2}{R_{\max}^2}\right) \\ &\leq 2B \left\lfloor \frac{K}{m_0} \right\rfloor \exp\left(-\frac{2n_{\min}\epsilon^2}{R_{\max}^2}\right). \end{aligned} \quad (3)$$

Setting the last expression equal to δ and solving for ϵ proves the stated bound. Finally, if selection does not distort the descendant reward law beyond the retained prefix u , then $V_{\text{sel}}(v) = \mathbb{E}[R \mid Y_{<p} = u] = V(u)$. $\square$

E.3 Proof of the prefix-matching claim

Theorem 6 (Collision-only prefix matching is shallow-biased). *Let $\mathcal{A} = \{p_1 < \dots < p_M\}$ be the admissible boundaries. For $p \in \mathcal{A}$, define $U_p = Y_{<p}$, $\kappa(p) = \sum_{u \in \mathcal{V}^p} \mathbb{P}(U_p = u)^2$, and $\eta(p) = \text{Var}(\mathbb{E}[R \mid U_p]) / \text{Var}(R)$ when $\text{Var}(R) > 0$. Then $\kappa(p)$ is non-increasing in p . Hence any collision-only selector $p_{\text{coll}} \in \arg \max_{p \in \mathcal{A}} \kappa(p)$ selects from the earliest plateau $\{p \in \mathcal{A} : \kappa(p) = \kappa(p_1)\}$; if $\kappa(p_1) > \kappa(p_2)$, it uniquely selects p_1 . Moreover, there exist $p_0 < p_*$ such that $\kappa(p_0) > \kappa(p_*) > 0$, while $\eta(p_0) = 0$ and $\eta(p_*) = 1$.*

Proof. We first prove that $\kappa(p)$ is non-increasing. The prefix partition at depth $p+1$ refines the prefix partition at depth p . Fix a length- p prefix u , and let $q_a = \mathbb{P}(U_{p+1} = (u, a))$. Then $\mathbb{P}(U_p = u) = \sum_a q_a$, and

$$\sum_a q_a^2 \leq \left(\sum_a q_a\right)^2 = \mathbb{P}(U_p = u)^2.$$

Summing over all $u \in \mathcal{V}^p$ gives $\kappa(p+1) \leq \kappa(p)$. Thus, by induction, $p_a < p_b$ implies $\kappa(p_a) \geq \kappa(p_b)$.

It follows that any maximizer of $\kappa(p)$ over $\mathcal{A} = \{p_1 < \dots < p_M\}$ lies in the earliest plateau $\{p \in \mathcal{A} : \kappa(p) = \kappa(p_1)\}$. If $\kappa(p_1) > \kappa(p_2)$, then no deeper admissible boundary can match $\kappa(p_1)$, so p_1 is the unique maximizer.

It remains to show that high collision need not imply reward information. Consider two admissible boundaries $p_0 < p_*$. Suppose all responses

share the same prefix at p_0 . Then U_{p_0} is deterministic, so $\kappa(p_0) = 1$. Since $\mathbb{E}[R \mid U_{p_0}]$ is constant, $\eta(p_0) = 0$.

At boundary p_* , suppose there are two prefixes u_0, u_1 , each with probability $1/2$, and let $R = \mathbf{1}\{U_{p_*} = u_1\}$. Then $\kappa(p_*) = (1/2)^2 + (1/2)^2 = 1/2$, so $\kappa(p_0) > \kappa(p_*) > 0$. Also, $\mathbb{E}[R \mid U_{p_*} = u_0] = 0$ and $\mathbb{E}[R \mid U_{p_*} = u_1] = 1$, so $\mathbb{E}[R \mid U_{p_*}] = R$. Hence $\text{Var}(\mathbb{E}[R \mid U_{p_*}]) = \text{Var}(R)$, and $\eta(p_*) = 1$. Therefore collision-only selection can prefer a shallower boundary with larger collision but zero reward informativeness. $\square$