BASED: BRANCH-AWARE SOFT EMBEDDING DISTILLATION FOR REASONING LLMS

Anonymous authors
Paper under double-blind review

ABSTRACT

On-policy distillation (OPD) provides dense teacher supervision along student-generated reasoning trajectories, but each discrete rollout follows only one continuation at a reasoning branch, leaving alternative plausible branches unsupervised. Soft embeddings offer a natural way to represent multiple candidate branches within a continuous trajectory, yet we find that naive soft-trajectory distillation loses branch-specific information and may lead to branch collapse. To explain this limitation, we introduce the Hard-Branch KL objective and derive an exact decomposition into a Mixture KL term and a Branch-Posterior KL term. We show that soft-trajectory KL approximates the Mixture KL while omitting branch-posterior supervision. This reveals that soft distillation preserves mixture-level supervision but discards the branch identities needed to distinguish alternative reasoning paths. Motivated by this analysis, we propose Branch-Aware Soft Embedding Distillation (BASED), which combines soft KL with a local branch-posterior correction to efficiently recover the missing branch-specific supervision. Experiments across eight reasoning benchmarks show that BASED improves Pass@1 over vanilla OPD across both student scales and achieves higher average Pass@8, while ablations and trajectory-diversity analyses provide further evidence that BASED helps preserve diverse reasoning branches.

1 INTRODUCTION

Knowledge distillation provides an effective way to transfer the capabilities of large language models (LLMs) to smaller models. For autoregressive generation, however, conventional distillation is often performed on fixed ground-truth or teacher-generated sequences, creating a mismatch between the contexts seen during training and those encountered when the student generates from its own predictions. Generalized Knowledge Distillation (GKD) addresses this issue through on-policy distillation (OPD), where the teacher provides dense token-level supervision along student-generated trajectories (Agarwal et al., 2024). This formulation is particularly attractive for long-form reasoning, where small early deviations can lead to substantially different downstream contexts. Recent work has further studied objectives and sampling strategies for LLM distillation, including reverse-KL-based distillation and uncertainty-aware OPD (Gu et al., 2024; Ko et al., 2024; Jin et al., 2026).

Despite being on-policy, standard OPD remains sparse with respect to alternative reasoning continuations. At each visited prefix, token-level KL already supervises the full next-token distribution, but once a token is sampled, subsequent teacher supervision is conditioned only on the resulting continuation. Other plausible choices may lead to distinct downstream reasoning states that are not visited along that trajectory. Sampling additional trajectories can broaden this coverage, but each rollout still resolves branching choices into a single realized path. Thus, the limitation is not the absence of next-token distributional supervision, but the lack of direct supervision over the downstream states induced by alternative branches.

Soft continuous reasoning offers a natural way to represent such alternatives without immediately committing to a single discrete continuation. Soft Thinking replaces a discrete token with a probability-weighted combination of candidate token embeddings, allowing information from multiple possible continuations to propagate through a single trajectory (Zhang et al., 2025). Related continuous-reasoning approaches likewise show that latent or soft representations can encode information beyond a single discrete chain of thought (Hao et al., 2025; Xu et al., 2025). More recently,

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.

SeLaR applies soft embeddings selectively at high-entropy reasoning states and shows that vanilla soft reasoning can collapse prematurely toward dominant tokens (Fu & Luo, 2026). These results suggest that soft embeddings could provide a natural surrogate for representing multiple reasoning branches during OPD.

However, aggregating branches into a soft state does not preserve all of the supervision available from evaluating the branches explicitly. First, because the model is nonlinear, the prediction produced from a weighted embedding is only an approximation to the mixture of predictions from the corresponding discrete branches. More importantly, even perfect mixture matching does not imply that individual branches are matched correctly: aggregation can preserve the overall predictive distribution while losing information about which branch produces which behavior. We formalize this observation through an idealized **Hard-Branch KL** objective and show that, under a shared branch prior, it decomposes exactly into a **Mixture KL** term and a **Branch-Posterior KL** term. This decomposition reveals a key limitation of soft-only distillation: soft trajectories can approximate mixture-level supervision, but they do not explicitly preserve branch identity.

Motivated by this analysis, we introduce **Branch-Aware Soft Embedding Distillation (BASED)**. BASED uses soft embeddings at uncertain reasoning states to efficiently aggregate plausible continuations, while supplementing soft-trajectory distillation with a lightweight **branch-posterior correction** that restores the branch-specific information lost by aggregation. Rather than constructing a full reasoning tree or generating many complete rollouts, BASED performs targeted branch-aware supervision only around selected branching states. In this way, it retains the advantages of soft trajectories while recovering supervision that soft-only distillation omits.

We evaluate BASED on eight mathematical and scientific reasoning benchmarks. Across different student scales, BASED improves Pass@1 over vanilla OPD on all eight benchmarks and achieves higher average Pass@8. Ablation studies further examine the contribution of local branch-posterior correction and the effects of the entropy threshold, mixture temperature, and target-selection policy. Finally, trajectory-diversity analyses compare vanilla OPD, soft-only distillation, and BASED using Self-BLEU and chrF-based clustering, providing additional evidence that branch-aware supervision helps preserve diverse reasoning trajectories.

Our contributions are threefold:

- **We revisit OPD from a novel branch-level perspective.** We formulate an idealized Hard-Branch KL objective and derive an exact decomposition into mixture-level and branch-posterior matching, providing a formal view of supervision over alternative reasoning continuations.
- **We study soft embeddings in the context of OPD and identify a theoretical limitation.** We show that soft-trajectory KL approximates mixture-level matching but does not explicitly preserve branch-posterior information, revealing how soft aggregation can lose branch identity.
- **We propose Branch-Aware Soft Embedding Distillation (BASED).** BASED complements soft-trajectory distillation with local branch-posterior correction. Across eight reasoning benchmarks and two student scales, it improves Pass@1 over vanilla OPD on every benchmark and achieves higher average Pass@8, with ablations and trajectory-diversity analyses further supporting the role of branch-posterior supervision.

2 PRELIMINARIES

2.1 ON-POLICY DISTILLATION

On-policy distillation (OPD) mitigates the train–inference distribution mismatch of conventional knowledge distillation by training the student on trajectories sampled from its current policy (Agarwal et al., 2024). Given a prompt $x \sim \mathcal{D}$, the student generates a trajectory $y = (y_1, \ldots, y_L)$. Let $p_T^t = p_T(\cdot \mid x, y_{<t})$ and $p_\theta^t = p_\theta(\cdot \mid x, y_{<t})$ be the next-token distributions produced by the teacher and student. The forward-KL objective is

$$\mathcal{L}_{\mathrm{OPD}}(\theta) = \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{y\sim \mathrm{sg}[p_\theta(\cdot|x)]}\left[\frac{1}{L}\sum_{t=1}^{L} D_{\mathrm{KL}}(p_T^t \| p_\theta^t)\right].$$

where $\mathrm{sg}[\cdot]$ denotes the stop-gradient operation. We focus on **GKD-style OPD**, where the token-level divergence between the teacher and student distributions is directly optimized as the training loss. This direct divergence minimization differs from PG-style OPD, which converts the divergence into a reward or advantage and updates the student through a policy-gradient objective. OPD can also be instantiated with other distributional divergences, such as reverse KL and generalized Jensen–Shannon divergence.

2.2 Soft Embeddings

Standard autoregressive decoding commits to a single discrete token at each step and feeds its embedding back into the model. Soft embedding methods instead retain multiple candidate tokens through a probability-weighted combination of their embeddings (Zhang et al., 2025; Fu & Luo, 2026).

Let $z_t \in \mathbb{R}^{|\mathcal{V}|}$ denote the logits at step t, and let $\mathcal{V}_t^k$ contain the top-k tokens with the largest logits. We denote by $q_t(v)$ the corresponding probabilities renormalized over $\mathcal{V}_t^k$. The corresponding soft embedding is

$$\tilde{e}_t = \sum_{v \in \mathcal{V}_t^k} q_t(v) E_v,$$

where E_v is the embedding of token v. Instead of feeding the embedding of a single sampled token into the next autoregressive step, the model consumes the soft embedding $\tilde{e}_t$, allowing multiple candidates influence subsequent computation within one continuous trajectory.

3 Theoretical Analysis

We analyze how branch aggregation changes distillation supervision. We first define a shared-prior Hard-Branch KL objective, then decompose it into mixture-level and branch-posterior matching, and finally relate soft-trajectory KL to the mixture term. We focus on forward KL; Appendix A.3 gives the reverse-KL counterpart.

3.1 Hard-Branch KL Objective

Assume the teacher and student share a tokenizer and vocabulary $\mathcal{V}$. Fix a prompt x and a prediction position t. Let $\mathcal{H}$ be a finite family of hard branch histories, where each $h \in \mathcal{H}$ specifies a discrete prefix $y_{<t}^{(h)}$. Let $\alpha_h \geq 0$, with $\sum_h \alpha_h = 1$, denote the student-induced probability assigned to branch h. Both models are evaluated under this shared branch prior, with the branch histories and weights fixed during optimization. This does not require their independently generated prefix distributions to agree.

Writing $T_h = p_T(\cdot \mid x, y_{<t}^{(h)})$ and $S_h = p_\theta(\cdot \mid x, y_{<t}^{(h)})$, define the **Hard-Branch KL** objective

$$\mathcal{L}_{\mathrm{hard}}^F = \mathbb{E}_{h \sim \mathrm{sg}[\alpha]} \left[D_{\mathrm{KL}}(T_h \| S_h) \right] = \sum_{h \in \mathcal{H}} \alpha_h D_{\mathrm{KL}}(T_h \| S_h).$$

This objective matches teacher and student predictions separately on each branch. Given independent samples $h_1, \ldots, h_K \sim \alpha$, the average $K^{-1} \sum_i D_{\mathrm{KL}}(T_{h_i} \| S_{h_i})$ is an unbiased Monte Carlo estimator. Thus, finite-rollout OPD faces a branch-coverage limitation, not an intrinsic omission of branches from its population objective. Explicit enumeration avoids sampling error, but a family with m branching positions and k candidates per position contains k^m histories. This motivates an aggregated surrogate for branch-wise supervision.

3.2 Exact Decomposition: Mixture KL + Branch Posterior

Define the joint distributions $P_T(h, v) = \alpha_h T_h(v)$ and $P_S(h, v) = \alpha_h S_h(v)$, with next-token marginals

$$T_{\mathrm{mix}}(v) = \sum_h \alpha_h T_h(v), \qquad S_{\mathrm{mix}}(v) = \sum_h \alpha_h S_h(v).$$

Proposition 1 (Exact Hard-Branch KL decomposition). If the branch-conditioned distributions are strictly positive on the finite vocabulary, then under a shared branch prior,

$$\mathcal{L}^F_{\text{hard}} = \underbrace{D_{\text{KL}}(T_{\text{mix}} \| S_{\text{mix}})}_{\mathcal{L}^F_{\text{mix}}} + \underbrace{\mathbb{E}_{v \sim T_{\text{mix}}} \left[D_{\text{KL}}(P_T(\cdot \mid v) \| P_S(\cdot \mid v)) \right]}_{\mathcal{L}^F_{\text{post-full}}},$$

where, for branches with positive prior mass,

$$P_T(h \mid v) = \frac{\alpha_h T_h(v)}{T_{\text{mix}}(v)}, \qquad P_S(h \mid v) = \frac{\alpha_h S_h(v)}{S_{\text{mix}}(v)}.$$

The result follows by applying the relative-entropy chain rule to $\mathcal{L}^F_{\text{hard}} = D_{\text{KL}}(P_T \| P_S)$ (Cover & Thomas, 2006, Theorem 2.5.3). A full derivation is provided in Appendix A.1.

The mixture term compares aggregate next-token predictions after the branch identity has been marginalized out. In contrast, $P_M(h \mid v)$ describes the relative responsibility of branch h for a candidate next token v, under the shared prior. The posterior term therefore aligns the teacher's and student's **branch-to-token associations** across the full vocabulary.

For example, consider two equally weighted branches over two tokens:

$$T_1 = (0.9, 0.1), \quad T_2 = (0.1, 0.9), \qquad S_1 = S_2 = (0.5, 0.5).$$

Both mixtures equal $(0.5, 0.5)$, so $\mathcal{L}^F_{\text{mix}} = 0$, yet $\mathcal{L}^F_{\text{post-full}} > 0$: the student loses branch distinctions despite matching the mixture. This illustrates an ambiguity of mixture supervision.

3.3 Soft KL as an Approximation to Mixture KL

We now relate the mixture term in Section 3.2 to distillation with probability-weighted embeddings. Consider a set of branching positions $\mathcal{A}_{<t}$ in the prefix before prediction position t. At each $s \in \mathcal{A}_{<t}$, let $\mathcal{V}^k_s$ contain k candidate tokens, with normalized weights $q_s(v)$ for $v \in \mathcal{V}^k_s$. Keeping all non-branching positions fixed, these choices induce the hard-branch family $\mathcal{H} = \prod_{s \in \mathcal{A}_{<t}} \mathcal{V}^k_s$, where each branch h selects one candidate h_s at every branching position. We assign it the product weight $\alpha_h = \prod_{s \in \mathcal{A}_{<t}} q_s(h_s)$, defining the shared branch prior.

For $M \in \{T, \theta\}$, let $\mathbf{e}_M(h)$ denote the prefix-embedding sequence of branch h, and view $p_M(\cdot \mid x, \mathbf{e})$ as the next-token distribution produced when the prefix is represented by embedding sequence $\mathbf{e}$. Define the weighted mean prefix embedding

$$\bar{\mathbf{e}}_M = \sum_{h \in \mathcal{H}} \alpha_h \mathbf{e}_M(h), \qquad [\bar{\mathbf{e}}_M]_s = \sum_{v \in \mathcal{V}^k_s} q_s(v) E_M[v] \quad (s \in \mathcal{A}_{<t}),$$

while non-branching positions remain unchanged. Thus, $\bar{\mathbf{e}}_M$ is exactly the prefix obtained by replacing each branching position with the corresponding probability-weighted embedding, using model M's own embedding matrix.

Following the probability-level first-order approximation in Soft Thinking (Zhang et al., 2025, Section 3.3), we linearize $p_M(\cdot \mid x, \mathbf{e})$ around $\bar{\mathbf{e}}_M$. Because $\bar{\mathbf{e}}_M$ is the weighted mean of the branch embeddings, the first-order deviations cancel after averaging over $h \sim \alpha$, giving

$$\sum_h \alpha_h \, p_M(\cdot \mid x, \mathbf{e}_M(h)) \approx p_M(\cdot \mid x, \bar{\mathbf{e}}_M).$$

Define the predictions from the probability-weighted prefixes as $T_{\text{soft}} = p_T(\cdot \mid x, \bar{\mathbf{e}}_T)$ and $S_{\text{soft}} = p_\theta(\cdot \mid x, \bar{\mathbf{e}}_\theta)$. Meanwhile, by Section 3.2, the corresponding branch mixtures are $T_{\text{mix}} = \sum_h \alpha_h p_T(\cdot \mid x, \mathbf{e}_T(h))$ and $S_{\text{mix}} = \sum_h \alpha_h p_\theta(\cdot \mid x, \mathbf{e}_\theta(h))$. Therefore, the first-order approximation implies

$$T_{\text{soft}} \approx T_{\text{mix}}, \qquad S_{\text{soft}} \approx S_{\text{mix}}.$$

Under the approximation conditions detailed in Appendix A.2, these distribution-level approximations yield

$$\mathcal{L}^F_{\text{soft}} := D_{\text{KL}}(T_{\text{soft}} \| S_{\text{soft}}) \approx D_{\text{KL}}(T_{\text{mix}} \| S_{\text{mix}}) = \mathcal{L}^F_{\text{mix}}.$$

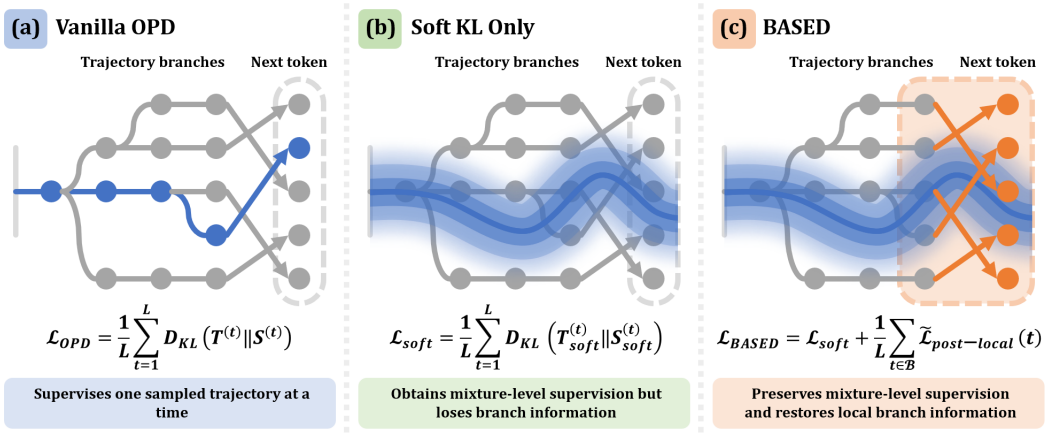


Figure 1: Comparison of branch supervision in vanilla OPD, soft KL, and BASED. Vanilla OPD supervises a sampled hard trajectory, soft KL aggregates multiple branches at the mixture level, and BASED additionally applies local branch-posterior correction.

Combining this with Proposition 1 gives

$$\mathcal{L}^F_{\text{hard}} = \mathcal{L}^F_{\text{mix}} + \mathcal{L}^F_{\text{post-full}} \approx \mathcal{L}^F_{\text{soft}} + \mathcal{L}^F_{\text{post-full}}.$$

Soft KL therefore approximates only the mixture-matching component of Hard-Branch KL; it does not explicitly match the teacher's and student's branch posteriors. This motivates the tractable branch-posterior correction introduced in Section 4. Formal approximation conditions and error bounds are provided in Appendix A.2; the relation above is an objective-level approximation rather than a general equality or gradient equivalence.

4 Method

Motivated by the decomposition in Section 3, **Branch-Aware Soft Embedding Distillation (BASED)** combines KL distillation on entropy-gated soft trajectories with a local branch-posterior correction (Figure 1). The former provides mixture-level supervision, while the latter restores branch-conditioned information that is not explicitly matched by soft KL. We present the forward-KL formulation; Appendix B.3 gives its reverse-KL counterpart.

4.1 Entropy-Gated Soft Trajectory Rollout

We selectively activate soft embeddings (Zhang et al., 2025; Fu & Luo, 2026) along student-generated trajectories. At generation step t, let z_t be the student's logits conditioned on the prompt and current mixed prefix, and let $\mathcal{V}^k_t$ contain the k tokens with the largest logits, where $k \geq 2$. We define the untempered top-k distribution $r_t(v)$ and its normalized entropy $\bar{H}_t$

$$r_t(v) = \frac{\exp z_t(v)}{\sum_{w\in\mathcal{V}^k_t} \exp z_t(w)}, \qquad \bar{H}_t = -\frac{1}{\log k} \sum_{v\in\mathcal{V}^k_t} r_t(v) \log r_t(v).$$

When $\bar{H}_t \leq \tau$, we sample a hard token y_t using the discrete sampler described in Appendix B.1. Otherwise, t becomes a **soft anchor**. In this case, we apply mixture temperature $\beta > 0$ only to construct the soft-mixture weights,

$$q_t(v) = \frac{\exp(z_t(v)/\beta)}{\sum_{w\in\mathcal{V}^k_t} \exp(z_t(w)/\beta)}, \qquad v \in \mathcal{V}^k_t.$$

The student then feeds back $E_\theta[y_t]$ at a hard position and $\sum_{v\in\mathcal{V}^k_t} q_t(v) E_\theta[v]$ at a soft anchor. Thus, τ controls whether a soft anchor is activated, whereas β controls only the concentration of its mixture weights.

The trajectory record $\mathcal{R}$ stores the hard tokens, anchor positions, candidate token IDs and weights, and the normalized entropy of each generated position. Rollout generation is gradient-free, and the recorded token choices, candidate sets, and mixture weights are treated as fixed during the subsequent update.

4.2 Soft KL Distillation

For a record with L generated positions, let $\mathcal{A} \subseteq \{1, \dots, L\}$ denote the set of soft anchors. Each model $M \in \{T, \theta\}$ reconstructs the recorded trajectory using its own embedding matrix, with $e_t^M = E_M[y_t]$ for $t \notin \mathcal{A}$ and $e_t^M = \sum_{v \in \mathcal{V}_t^k} q_t(v) E_M[v]$ for $t \in \mathcal{A}$. The two models therefore share token identities and mixture weights, but not embedding vectors, allowing their hidden dimensions to differ.

Let $\bar{\mathbf{e}}_{M,<t}$ denote the reconstructed prefix before prediction position t, including the prompt and preceding generated inputs. Consistent with Section 3.3, define $T_{\text{soft}}^{(t)} = p_T(\cdot \mid x, \bar{\mathbf{e}}_{T,<t})$ and $S_{\text{soft}}^{(t)} = p_\theta(\cdot \mid x, \bar{\mathbf{e}}_{\theta,<t})$. We apply KL distillation at all generated prediction positions, including both hard-token and soft-anchor positions:

$$\mathcal{L}_{\text{soft}}^F(x, \mathcal{R}) = \frac{1}{L} \sum_{t=1}^{L} D_{\text{KL}}\Big(T_{\text{soft}}^{(t)} \| S_{\text{soft}}^{(t)}\Big).$$

The teacher remains frozen, while gradients propagate through the student's replay computations. As shown in Section 3.3, this objective approximates mixture-level matching but does not explicitly enforce branch-posterior agreement.

4.3 Local Branch-Posterior Correction

We therefore supplement soft KL with local branch-posterior supervision at selected positions. Rather than expanding all combinations of earlier branch choices, we resolve only the immediately preceding soft anchor while retaining all earlier inputs in their recorded mixed form.

Target selection. Let the ordered anchor positions be $\mathcal{A} = \{a_1 < \cdots < a_m\}$. Following the eligibility convention in Appendix B.1, the candidate target positions are $\mathcal{C} = \{a_j + 1 : 1 \le j < m\}$. We select up to N_{post} targets with the largest recorded target entropies,

$$\mathcal{B} = \text{TopN}_{t \in \mathcal{C}}\big(\bar{H}_t; N_{\text{post}}\big).$$

Importantly, the ranking uses the entropy at the target position t, rather than the entropy of its preceding anchor.

Local branch construction. For each $t \in \mathcal{B}$, let $b = t - 1$ be its preceding anchor. For every $v \in \mathcal{V}_b^k$, we replace only the reconstructed input at b with the hard embedding $E_M[v]$, leaving all earlier inputs unchanged. Let $\bar{\mathbf{e}}_{M,<t}^{b \to v}$ denote the resulting prefix, and define the branch-conditioned predictions as $T_v^{(t)} = p_T(\cdot \mid x, \bar{\mathbf{e}}_{T,<t}^{b \to v})$ and $S_v^{(t)} = p_\theta(\cdot \mid x, \bar{\mathbf{e}}_{\theta,<t}^{b \to v})$.

Using the same detached branch prior q_b for both models, the local mixture distributions are $T_{\text{mix,local}}^{(t)}(u) = \sum_{v \in \mathcal{V}_b^k} q_b(v) T_v^{(t)}(u)$ and analogously $S_{\text{mix,local}}^{(t)}$. Their branch posteriors are

$$P_T^{(t)}(v \mid u) = \frac{q_b(v) T_v^{(t)}(u)}{T_{\text{mix,local}}^{(t)}(u)}, \qquad P_S^{(t)}(v \mid u) = \frac{q_b(v) S_v^{(t)}(u)}{S_{\text{mix,local}}^{(t)}(u)}.$$

We then define the local posterior correction

$$\widetilde{\mathcal{L}}_{\text{post-local}}^F(t) = \sum_{u \in \mathcal{V}} T_{\text{mix,local}}^{(t)}(u) D_{\text{KL}}\Big(P_T^{(t)}(\cdot \mid u) \| P_S^{(t)}(\cdot \mid u)\Big).$$

The correction therefore matches branch-to-token associations over the **full target vocabulary**, rather than only the sampled token. Because only the immediately preceding anchor is expanded, it provides local branch-posterior supervision without constructing the full-history posterior term in Section 3.2.

Training objective. Both components are normalized by the total number L of generated positions:

$$\mathcal{L}^{F}_{\mathrm{BASED}}(x, \mathcal{R}) = \mathcal{L}^{F}_{\mathrm{soft}}(x, \mathcal{R}) + \frac{1}{L} \sum_{t \in \mathcal{B}} \widetilde{\mathcal{L}}^{F}_{\text{post-local}}(t).$$

Let $R_\theta(\cdot \mid x)$ denote the student-induced distribution over trajectory records. The overall objective is

$$\mathcal{L}^{F}_{\mathrm{BASED}}(\theta) = \mathbb{E}_{x \sim \mathcal{D}} \mathbb{E}_{\mathcal{R} \sim \mathrm{sg}[R_\theta(\cdot|x)]} \left[\mathcal{L}^{F}_{\mathrm{BASED}}(x, \mathcal{R}) \right].$$

The local correction requires at most kN_{post} branch-conditioned predictions per model and trajectory, without expanding a full branch tree. Further implementation and efficiency details are provided in Appendix B.2. BASED modifies only training; inference uses standard discrete decoding without soft inputs or branch expansion.

Table 1: Main results (%) with Qwen3-8B in non-thinking mode as the teacher. AVG is the unweighted average over the eight benchmarks. Bold indicates the best result within each student–metric block, including ties.

Method	MATH 500	AIME 2024	AIME 2025	AIME 2026	HMMT Feb. 26	AMC 2023	Minerva Math	GPQA-D	**AVG**
Student: Qwen3-1.7B-Base									
					Pass@1				
Base	17.48	1.15	1.25	0.63	0.38	10.23	8.59	6.50	5.78
+ OPD	59.75	3.85	3.44	3.33	1.33	30.63	25.32	24.49	19.02
+ **BASED**	**60.78**	**6.77**	**4.58**	**4.79**	**2.46**	**33.52**	**26.52**	**25.13**	**20.57**
					Pass@8				
Base	62.00	7.81	8.72	4.81	2.13	43.44	34.56	37.88	25.17
+ OPD	**86.40**	16.41	**15.92**	13.76	6.88	63.17	52.21	78.79	41.69
+ **BASED**	84.00	**20.84**	13.57	**15.57**	**7.85**	**66.72**	**52.94**	**80.81**	**42.79**
Student: Qwen3-4B-Base									
					Pass@1				
Base	55.25	6.56	6.25	5.73	3.13	33.28	18.75	8.78	17.22
+ OPD	77.75	15.52	13.13	11.67	4.83	54.69	43.06	38.01	32.33
+ **BASED**	**78.93**	**17.60**	**13.96**	**13.02**	**7.20**	**56.25**	**44.90**	**40.03**	**33.98**
					Pass@8				
Base	89.60	24.33	25.24	16.70	9.33	76.72	51.10	46.97	42.50
+ OPD	**93.00**	33.63	33.38	28.58	18.91	**87.62**	65.07	85.86	55.76
+ **BASED**	**93.00**	**38.07**	**33.74**	**29.49**	**22.53**	85.31	**68.01**	**90.91**	**57.63**

5 Experiments

5.1 Experimental Settings

Models and training data. We use Qwen3-8B in non-thinking mode as the teacher and Qwen3-1.7B-Base and Qwen3-4B-Base as the students (Yang et al., 2025). All students are trained on problems drawn from DAPO-Math-17k (Yu et al., 2025). We compare BASED with the corresponding base models and vanilla on-policy distillation (OPD) (Agarwal et al., 2024). Additional training details are provided in Appendix C.

Evaluation. We evaluate on seven mathematics benchmarks: MATH-500 (Hendrycks et al., 2021), AIME 2024, AIME 2025, AIME 2026, HMMT February 2026, AMC 2023, and MinervaMath (Lewkowycz et al., 2022). We also evaluate scientific reasoning on GPQA-Diamond (GPQA-D) (Rein et al., 2024). We report Pass@1 and Pass@8, sampling eight responses per problem for MATH-500, MinervaMath, and GPQA-D, and 32 for each of the remaining benchmarks. AVG denotes the unweighted average across the eight benchmarks. Further evaluation details are provided in Appendix C.3.

Table 2: Ablation results (%) with Qwen3-1.7B-Base as the student. AVG is the unweighted average over all eight benchmarks. The first row is the default configuration. Bold marks the highest Pass@1 and Pass@8 in each result column.

Method	Entropy threshold	β	Target selection policy	Pass@1 / Pass@8		
				AIME 2026	GPQA-D	AVG
BASED	0.7	1.0	Immediate next token	**4.79 / 15.57**	25.13 / 80.81	**20.57** / 42.79
Soft only	0.7	1.0	—	3.33 / 12.79	23.23 / 76.77	20.07 / 42.78
BASED	0.5	1.0	Immediate next token	3.33 / 14.63	26.52 / **83.33**	19.93 / **43.60**
BASED	0.7	0.7	Immediate next token	3.13 / 10.50	24.94 / 77.27	19.82 / 41.71
BASED	0.7	1.5	Immediate next token	3.65 / 13.75	**26.58** / 81.31	20.50 / 42.55
BASED	0.7	1.0	Next soft anchor	3.96 / 13.34	25.13 / 81.82	20.46 / 43.20

5.2 Main Results

Table 1 shows that both OPD and BASED substantially improve over the initial students. More importantly, BASED improves Pass@1 over vanilla OPD on all eight benchmarks at both student scales. The macro-average increases from 19.02 to 20.57 for the 1.7B student and from 32.33 to 33.98 for the 4B student, corresponding to gains of 1.55 and 1.65 percentage points, respectively. These gains also extend to GPQA-D, rather than being confined to mathematics.

BASED also improves macro-average Pass@8 by 1.09 and 1.88 percentage points for the 1.7B and 4B students, respectively. Across the 16 student–benchmark comparisons, it improves Pass@8 in 12 cases, ties in one, and decreases in three. Thus, BASED provides consistent single-sample gains and higher average multi-sample performance. Additional results using reverse KL are provided in Appendix E.

5.3 Ablation Study

Table 2 examines four design choices. **Local posterior correction.** Adding the correction provides a modest gain in average Pass@1 over soft-only distillation while leaving average Pass@8 nearly unchanged, with improvements on both displayed benchmarks. This suggests that branch-level supervision provides complementary information beyond soft distribution matching. **Entropy threshold.** The two tested thresholds yield similar average performance, with a Pass@1/Pass@8 trade-off. The higher threshold more selectively introduces soft anchors at high-uncertainty positions. **Softness parameter β.** The default β achieves the best averages among the tested temperatures. A smaller β concentrates mixtures toward their top-1 tokens and reduces average performance, consistent with diminished benefits from soft branching. A larger β produces more diffuse mixtures with slightly lower averages, possibly reflecting increased mismatch with discrete-token trajectories. **Target selection.** Replacing the immediate-next-token target with the next soft anchor yields comparable averages, with slightly higher Pass@8 but lower Pass@1. We retain the immediate next token for its simpler, local supervision.

5.4 Trajectory Diversity Analysis

We examine whether branch-aware supervision helps preserve diversity among correct responses. For two AIME 2026 problems (Q4 and Q15), we sample 512 responses from each of vanilla OPD, soft only, and BASED. We measure Self-BLEU-4 (Zhu et al., 2018) on four randomly selected correct responses per method, averaging over 50 repetitions; lower scores indicate greater lexical diversity. We additionally follow the trajectory-level analysis of Matsutani et al. (2026), using chrF similarity and UPGMA hierarchical clustering separately for correct and incorrect responses. Detailed settings are provided in Appendix D.

As shown in Figure 2, BASED reduces Self-BLEU-4 relative to soft only on both Q4 ($53.45 \rightarrow 52.26$) and Q15 ($42.59 \rightarrow 41.56$). Compared with vanilla OPD, its score is slightly higher on Q4 (52.26 vs. 52.16) but lower on Q15 (41.56 vs. 42.20). Clustering provides complementary evidence: BASED yields more correct clusters and fewer incorrect clusters than either baseline on both problems. Relative to soft only, the number of correct clusters increases from 11 to 12 on Q4

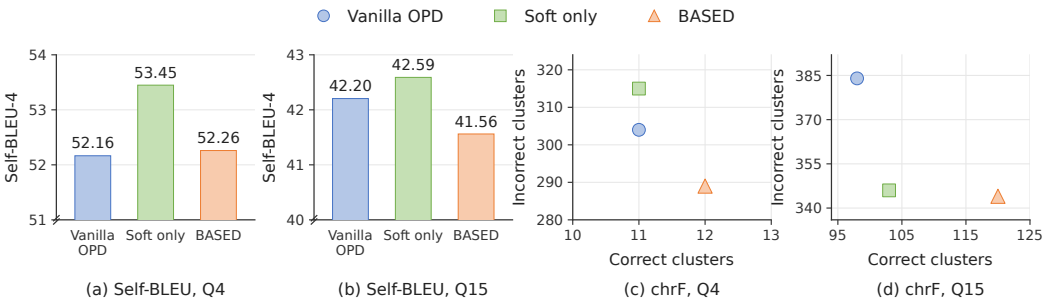


Figure 2: Trajectory diversity on two AIME 2026 problems. (a–b) Self-BLEU-4 of correct responses; lower is better. (c–d) Correct and incorrect cluster counts from chrF-based UPGMA clustering.

and from 103 to 120 on Q15, while incorrect clusters decrease from 315 to 289 and from 346 to 344, respectively. Together, these observations suggest that local branch-posterior correction helps preserve diversity among correct responses, consistent with the complementary branch-sensitive supervision it adds beyond soft distribution matching.

6 Related Work

On-Policy Distillation. Knowledge distillation for autoregressive language models increasingly incorporates student-generated sequences to reduce the discrepancy between training and inference distributions. MiniLLM optimizes reverse KL on student-generated outputs to mitigate exposure bias (Gu et al., 2024), while DistiLLM combines student-generated samples with a skew-KL objective and adaptive off-policy sampling for more efficient distillation (Ko et al., 2024). GKD formalizes on-policy distillation by querying the teacher for dense token-level supervision along trajectories generated by the student (Agarwal et al., 2024). More recently, Entropy-Aware On-Policy Distillation studies diversity degradation under reverse KL and selectively incorporates forward KL at high-entropy teacher states to better preserve plausible alternatives (Jin et al., 2026). Unlike these methods, which primarily modify the supervision applied along sampled trajectories, BASED focuses on efficiently preserving supervision from alternative branch-conditioned continuations.

Soft Embeddings and Continuous Reasoning. Recent work explores continuous representations as an alternative to committing to a discrete token at every reasoning step. Coconut recurrently feeds hidden states back into the model and finds that continuous thoughts can represent multiple potential future reasoning steps (Hao et al., 2025), while CODI compresses explicit chain-of-thought into continuous representations through self-distillation (Shen et al., 2025), and SoftCoT introduces learned soft thought representations for efficient continuous reasoning (Xu et al., 2025). Most closely related to our work, Soft Thinking constructs continuous reasoning tokens as probability-weighted mixtures of token embeddings, allowing multiple candidate semantics to be propagated within a single trajectory (Zhang et al., 2025), and SeLaR selectively applies such soft embeddings at high-entropy states while addressing their tendency to collapse toward the dominant token (Fu & Luo, 2026). BASED differs in using soft embeddings as a computational surrogate for multi-branch distillation and explicitly compensating for the branch-specific information lost during aggregation.

7 Conclusion

We revisited on-policy distillation from a branch-level perspective and showed that soft-trajectory KL captures mixture-level supervision while omitting explicit branch-posterior matching. Motivated by this analysis, we proposed BASED, which augments entropy-gated soft-trajectory distillation with lightweight local branch-posterior correction. Across eight reasoning benchmarks and two student scales, BASED consistently improves Pass@1 over vanilla OPD and achieves higher average Pass@8, supporting branch-aware supervision as a promising direction for reasoning-model distillation.

AI USE STATEMENT

In this work, we used generative AI tools to assist with writing and language polishing, literature retrieval and discovery, experimental code generation, and mathematical derivations. AI-assisted writing was carefully reviewed and edited by the authors. Literature identified with AI assistance was manually verified against the original sources for relevance and factual accuracy. AI-generated experimental code was reviewed by the authors, with the core implementation logic carefully inspected and validated before use. AI-assisted mathematical derivations were independently checked and re-derived by the authors. The authors take full responsibility for the final content of this work, including its text, citations, code, mathematical claims, and experimental results.

REFERENCES

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 21246–21263, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/5be69a584901a26c521c2b51e40a4c20-Paper-Conference.pdf.

Boxing Chen and Colin Cherry. A systematic comparison of smoothing techniques for sentence-level BLEU. In Ondřej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Christof Monz, Matt Post, and Lucia Specia (eds.), *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pp. 362–367, Baltimore, Maryland, USA, June 2014. Association for Computational Linguistics. doi: 10.3115/v1/W14-3346. URL https://aclanthology.org/W14-3346/.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code, 2021. URL https://arxiv.org/abs/2107.03374.

Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley-Interscience, second edition, 2006.

Renyu Fu and Guibo Luo. SeLaR: Selective latent reasoning in large language models. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7073–7087, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.320. URL https://aclanthology.org/2026.acl-long.320/.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. Minillm: Knowledge distillation of large language models. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 32694–32717, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/8ac015d409635f196f9e3e9dcfb9a94e-Paper-Conference.pdf.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason E Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=Itxz7S4Ip3.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. In J. Vanschoren and S. Yeung (eds.), *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper_files/paper/2021/file/be83ab3ecd0db773eb2dc1b0a17836a1-Paper-round2.pdf.

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.

Woogyeol Jin, Taywon Min, Yongjin Yang, Dennis Wei, Yi Zhou, Swanand Ravindra Kadhe, Nathalie Baracaldo, and Kimin Lee. Entropy-aware on-policy distillation of language models. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=J5i09faOOf.

Jongwoo Ko, Sungnyun Kim, Tianyi Chen, and Se-Young Yun. DistiLLM: Towards streamlined distillation for large language models. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 24872–24895. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/ko24c.html.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, SOSP '23, pp. 611–626, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400702297. doi: 10.1145/3600006.3613165. URL https://doi.org/10.1145/3600006.3613165.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. Solving quantitative reasoning problems with language models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 3843–3857. Curran Associates, Inc., 2022. doi: 10.52202/068431-0278. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/18abbeef8cfe9203fdf9053c9c4fe191-Paper-Conference.pdf.

Kohsei Matsutani, Shota Takashiro, Gouki Minegishi, Takeshi Kojima, Yusuke Iwasawa, and Yutaka Matsuo. Rl squeezes, sft expands: A comparative study of reasoning llms. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 141482–141526, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/e52554a70e0df57a0bea11d1eca0c9b5-Paper-Conference.pdf.

Maja Popović. chrF: character n-gram F-score for automatic MT evaluation. In Ondřej Bojar, Rajan Chatterjee, Christian Federmann, Barry Haddow, Chris Hokamp, Matthias Huck, Varvara Logacheva, and Pavel Pecina (eds.), *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pp. 392–395, Lisbon, Portugal, September 2015. Association for Computational Linguistics. doi: 10.18653/v1/W15-3049. URL https://aclanthology.org/W15-3049/.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=Ti67584b98.

Zhenyi Shen, Hanqi Yan, Linhai Zhang, Zhanghao Hu, Yali Du, and Yulan He. CODI: Compressing chain-of-thought into continuous space via self-distillation. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 677–693, Suzhou, China, November

2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.36. URL https://aclanthology.org/2025.emnlp-main.36/.

Yige Xu, Xu Guo, Zhiwei Zeng, and Chunyan Miao. SoftCoT: Soft chain-of-thought for efficient reasoning with LLMs. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 23336–23351, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.1137. URL https://aclanthology.org/2025.acl-long.1137/.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, juncai liu, LingJun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Ru Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Yonghui Wu, and Mingxuan Wang. Dapo: An open-source llm reinforcement learning system at scale. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 113222–113244. Curran Associates, Inc., 2025. doi: 10.52202/085713-3775. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/a4277440d50f1f15d2cb4c14f7e0c0d2-Paper-Conference.pdf.

Zhen Zhang, Xuehai He, Weixiang Yan, Ao Shen, Chenyang Zhao, and Xin Wang. Soft thinking: Unlocking the reasoning potential of llms in continuous concept space. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 168990–169012. Curran Associates, Inc., 2025. doi: 10.52202/085713-5629. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/f7396d1c54d51416958d63e285377103-Paper-Conference.pdf.

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. Texygen: A benchmarking platform for text generation models. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, SIGIR '18, pp. 1097–1100, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450356572. doi: 10.1145/3209978.3210080. URL https://doi.org/10.1145/3209978.3210080.

A Additional Theoretical Details

A.1 Proof of Proposition 1

All sums below are over branches with positive prior mass and tokens in the finite vocabulary. Since the prior is shared,

$$D_{\mathrm{KL}}(P_T \| P_S) = \sum_{h,v} \alpha_h T_h(v) \log \frac{\alpha_h T_h(v)}{\alpha_h S_h(v)}$$
$$= \sum_h \alpha_h D_{\mathrm{KL}}(T_h \| S_h) = \mathcal{L}^F_{\mathrm{hard}}.$$

Alternatively, factoring the same joints by the next-token variable gives

$$\begin{aligned} D_{\mathrm{KL}}(P_T \| P_S) &= \sum_{h,v} P_T(h,v) \log \frac{T_{\mathrm{mix}}(v) P_T(h \mid v)}{S_{\mathrm{mix}}(v) P_S(h \mid v)} \\ &= \sum_{v} T_{\mathrm{mix}}(v) \log \frac{T_{\mathrm{mix}}(v)}{S_{\mathrm{mix}}(v)} \\ &\quad + \sum_{v} T_{\mathrm{mix}}(v) \sum_{h} P_T(h \mid v) \log \frac{P_T(h \mid v)}{P_S(h \mid v)} \\ &= \mathcal{L}^F_{\mathrm{mix}} + \mathcal{L}^F_{\text{post-full}}. \end{aligned}$$

This is the relative-entropy chain rule applied to the branch-token joint distributions (Cover & Thomas, 2006). Nonnegativity of conditional KL yields the lower bound and its equality condition. All identities hold pointwise in the prompt, prediction position, and frozen branch family, so taking a common expectation over these quantities preserves the decomposition.

The shared-prior qualification matters. With distinct priors α^T and α^S, the corresponding joint KL instead satisfies

$$D_{\mathrm{KL}}(\alpha_h^T T_h(v) \| \alpha_h^S S_h(v)) = D_{\mathrm{KL}}(\alpha^T \| \alpha^S) + \sum_h \alpha_h^T D_{\mathrm{KL}}(T_h \| S_h),$$

assuming compatible supports. The additional prior-matching term is not part of the shared-prior objective studied here. Freezing the common prior, in turn, specifies the GKD-style optimization semantics: no gradient is propagated through its construction.

A.2 Approximation Error and Sufficient Conditions

This appendix makes explicit the approximation conditions behind Section 3.3 and distinguishes soft-to-mixture approximation error from branch-posterior supervision. The bounds below are sufficient conditions, not assertions that the required constants are uniformly small for the trained models. For compactness, with x fixed as in Section 3.3, let $F_M(\mathbf{e}) := p_M(\cdot \mid x, \mathbf{e})$ for $M \in \{T, \theta\}$.

Taylor remainder and mixture predictions. Under the fixed branch ensemble of Section 3.3, write the expansion of each prediction map as

$$F_M(\mathbf{e}_M(h)) = F_M(\bar{\mathbf{e}}_M) + J_M(\bar{\mathbf{e}}_M)\big(\mathbf{e}_M(h) - \bar{\mathbf{e}}_M\big) + R_M(h),$$

where $R_M(h)$ is the Taylor remainder. Averaging over the shared prior cancels the linear term exactly, giving

$$T_{\mathrm{mix}} - T_{\mathrm{soft}} = \mathbb{E}_{h\sim\alpha}[R_T(h)], \qquad S_{\mathrm{mix}} - S_{\mathrm{soft}} = \mathbb{E}_{h\sim\alpha}[R_\theta(h)].$$

Thus, soft predictions approximate explicit mixtures when the averaged remainders are small. Equality holds for prediction maps that are affine over the relevant branch embeddings; differentiability alone does not ensure an accurate approximation.

Prediction error. Vectorize the prefix-embedding sequence. Suppose F_M is twice continuously differentiable on a neighborhood of the convex hull of $\{\mathbf{e}_M(h) : \alpha_h > 0\}$, and for every point z in that hull and direction d,

$$\|D^2 F_M(z)[d,d]\|_1 \le B_M \|d\|_2^2.$$

Writing $d_h = \mathbf{e}_M(h) - \bar{\mathbf{e}}_M$, Taylor's integral remainder is

$$R_M(h) = \int_0^1 (1-s) D^2 F_M(\bar{\mathbf{e}}_M + s d_h)[d_h, d_h]\, ds.$$

Consequently,

$$\left\| \sum_h \alpha_h F_M(\mathbf{e}_M(h)) - F_M(\bar{\mathbf{e}}_M) \right\|_1 \le \frac{B_M}{2} \sum_h \alpha_h \|\mathbf{e}_M(h) - \bar{\mathbf{e}}_M\|_2^2.$$

This bounds the prediction approximation by the product of local curvature and the weighted dispersion of the hard prefix embeddings. In the factorized ensemble of Section 3.3, the squared Euclidean dispersion of the concatenated prefix equals the sum of the position-wise embedding dispersions; the remaining hard positions contribute zero. Neither a high nor a low token entropy alone establishes a small value of this bound.

KL sensitivity. Set $p = T_{\text{mix}}$, $p' = T_{\text{soft}}$, $q = S_{\text{mix}}$, and $q' = S_{\text{soft}}$, and define

$$\delta_T = \|T_{\text{mix}} - T_{\text{soft}}\|_1, \qquad \delta_S = \|S_{\text{mix}} - S_{\text{soft}}\|_1.$$

Suppose all coordinates of these four probability vectors are at least $\eta > 0$, and use natural logarithms. For

$$G(p, q) = \sum_v p(v) \log \frac{p(v)}{q(v)},$$

its partial derivatives satisfy

$$\left|\frac{\partial G}{\partial p(v)}\right| = \left|1 + \log \frac{p(v)}{q(v)}\right| \le 1 + \log(1/\eta), \qquad \left|\frac{\partial G}{\partial q(v)}\right| = \frac{p(v)}{q(v)} \le \eta^{-1}.$$

Every point on the line segment from (p, q) to (p', q') obeys the same coordinate bound. Integrating the directional derivative along this segment gives

$$|G(p, q) - G(p', q')| \le \big(1 + \log(1/\eta)\big)\|p - p'\|_1 + \eta^{-1}\|q - q'\|_1,$$

and hence

$$\left|\mathcal{L}^F_{\text{mix}} - \mathcal{L}^F_{\text{soft}}\right| \le \big(1 + \log(1/\eta)\big)\delta_T + \eta^{-1}\delta_S.$$

Together with the curvature bound above, this provides sufficient conditions for the objective-level approximation in Section 3.3. Very small token probabilities can make this bound loose. The analysis therefore does not assert that small absolute probability error always yields a practically small KL error, nor does it provide a uniform approximation guarantee for arbitrary soft prefixes.

Approximation error versus missing supervision. Define the signed soft-to-mixture error as

$$\epsilon_{\text{soft}} = \mathcal{L}^F_{\text{mix}} - \mathcal{L}^F_{\text{soft}}.$$

Combining this definition with Proposition 1 gives the exact identity

$$\mathcal{L}^F_{\text{hard}} = \mathcal{L}^F_{\text{soft}} + \mathcal{L}^F_{\text{post-full}} + \epsilon_{\text{soft}}.$$

The first-order treatment in Section 3.3 omits ϵ_{soft}, but not the structurally distinct branch-posterior term. Even when $\epsilon_{\text{soft}} = 0$, matching the soft distributions does not generally match the branch posteriors. Because ϵ_{soft} can have either sign, soft KL itself is not necessarily a lower bound on Hard-Branch KL.

Scope of the analysis. These are objective-level statements. Approximation of the parameter gradients requires additional control of the parameter derivatives of the prediction maps and their remainders; it does not follow from closeness of the output distributions alone.

A.3 Reverse-KL Counterpart

Under the same shared prior, define

$$\mathcal{L}^R_{\text{hard}} = \sum_h \alpha_h D_{\text{KL}}(S_h \| T_h) = D_{\text{KL}}(P_S \| P_T).$$

Exchanging the two joint distributions in the chain-rule argument gives

$$\mathcal{L}^R_{\text{hard}} = \underbrace{D_{\text{KL}}(S_{\text{mix}} \| T_{\text{mix}})}_{\mathcal{L}^R_{\text{mix}}} + \underbrace{\mathbb{E}_{v \sim S_{\text{mix}}}\left[D_{\text{KL}}(P_S(h \mid v) \| P_T(h \mid v))\right]}_{\mathcal{L}^R_{\text{post-full}}}.$$

In particular, the outer expectation is over S_{mix}, not T_{mix}. With $\mathcal{L}^R_{\text{soft}} = D_{\text{KL}}(S_{\text{soft}} \| T_{\text{soft}})$ and $\epsilon^R_{\text{soft}} = \mathcal{L}^R_{\text{mix}} - \mathcal{L}^R_{\text{soft}}$,

$$\mathcal{L}^R_{\text{hard}} = \mathcal{L}^R_{\text{soft}} + \mathcal{L}^R_{\text{post-full}} + \epsilon^R_{\text{soft}}.$$

Under the same probability floor,

$$|\epsilon^R_{\text{soft}}| \leq \big(1 + \log(1/\eta)\big)\delta_S + \eta^{-1}\delta_T.$$

These KL decompositions are not asserted unchanged for arbitrary divergences.

B Implementation Details for BASED

B.1 Trajectory Records and Target Conventions

Detached records. Each generated position records its state type, readable token ID, and normalized entropy. A soft anchor additionally records its candidate token IDs and temperature-adjusted branch weights. The entropy is calculated from the untempered top-k distribution at every generated position, not only at anchors. The same recorded weights are used for the student's rollout input, both models' replay inputs, and the shared local branch prior. Teacher probabilities do not replace these weights.

At hard positions, the discrete sampler applies its rollout temperature and nucleus truncation over the aligned vocabulary; the soft-anchor top-k restriction does not constrain this sampler. At soft anchors, the readable token is selected using the top-1 policy, whereas subsequent model computation uses the mixed embedding. In the implementation, a sampling draw is made at every active generation step before selecting the readable token. A draw at a soft anchor is therefore discarded under the top-1 readable-token policy; this detail matters for reproducing the random-number stream.

Prediction indexing. Paper positions are one-based and code record indices are zero-based. A branch endpoint b is resolved before predicting $t = b + 1$. Neither the recorded token nor the recorded embedding at target position t is fed into its own counterfactual prediction. To obtain soft KL at all L generated positions, replay consumes the prompt and the first $L - 1$ generated input embeddings. Predictions include the final recorded position, including an end-of-sequence token when present; prompt tokens are not included in the loss denominator.

Eligibility. If the ordered soft anchors are $a_1, \ldots, a_m$, the implementation uses only $a_1, \ldots, a_{m-1}$ as branch endpoints, even when the final anchor has a following hard token in the record. This is an eligibility convention, not a requirement of the local posterior definition. Consequently, records with fewer than two soft anchors receive no correction. Candidates are ranked by the target's recorded $\bar{H}_t$. Equal scores are broken in favor of earlier target positions, and selected targets are processed in trajectory order. No entropy scores are recomputed on counterfactual branches.

Vocabulary and temperatures. Soft KL and local branch posteriors use the full aligned tokenizer vocabulary, excluding reserved output rows outside that vocabulary. Top-k controls the number of hard branch alternatives, not the vocabulary over which the posterior expectation is computed. The provided forward-KL configuration uses a distillation temperature of one, independently of the rollout sampling temperature and mixture temperature β.

B.2 Cached Local Evaluation and Exact Cache VJP

The implementation uses `exact_cache_vjp_seq` to avoid separately replaying the full prefix for each local branch. The teacher's and student's soft-trajectory forward passes first produce their master key/value caches. For a selected target $t = b + 1$, each cache is cropped to the prompt and generated inputs strictly before anchor b. The k candidate hard embeddings $E_M[v]$ are evaluated as a batch against this shared prefix. Because the target immediately follows the replaced anchor, each branch suffix contains one input token. Teacher computations are performed without gradients.

For the student, detaching a prefix cache and ignoring its derivatives would omit part of the local objective's parameter gradient. Instead, the cropped cache is copied into independent differentiable leaf tensors for each target. Backpropagation through the branch suffix computes both direct parameter gradients and cache cotangents. Cotangents are summed over the replicated branches and

accumulated into the corresponding positions of the master cache. A final vector–Jacobian product through the master prefix graph recovers the remaining prefix contribution, jointly with the soft-KL backward pass.

To express this precisely, let C_θ collect the student master-cache tensors, and let $g_t(\theta, C_\theta)$ denote the local posterior loss, with the target-specific cache crop left implicit. With the trajectory record fixed, the chain rule gives

$$\nabla_\theta \mathcal{L}^F_{\mathrm{BASED}}(x, \mathcal{R}) = \nabla_\theta \mathcal{L}^F_{\mathrm{soft}}(x, \mathcal{R}) + \frac{1}{L} \sum_{t \in \mathcal{B}} \left. \frac{\partial g_t(\theta, C)}{\partial \theta} \right|_{C=C_\theta} + (D_\theta C_\theta)^\top \left[\frac{1}{L} \sum_{t \in \mathcal{B}} \nabla_C g_t(\theta, C_\theta) \right].$$

The first sum of local derivatives is accumulated during the target-wise suffix backward passes, and the final term is evaluated through the shared master graph. Computations use deterministic evaluation-mode forward passes, with adapter dropout disabled. When the cached and uncached evaluations implement the same deterministic forward computation, this procedure recovers the gradient of the implemented soft-plus-local objective up to floating-point rounding. Here, *exact* concerns cache-gradient propagation; it does not assert that the local objective equals the full-history Hard-Branch KL.

Branch posteriors are evaluated directly in log space. For example, the teacher's log-joint probabilities are computed as $\log q_b(v) + \log T_v^{(t)}(u)$. The log token marginal is obtained by log-sum-exp over branches, and subtracting it from the log-joint probabilities yields the log posterior. The correction is then accumulated using the joint probabilities. This implements the conditional posterior expression directly rather than subtracting two nearly equal KL values.

B.3 Reverse-KL Formulation

The same rollout, target selection, local branches, and detached priors can be used with reverse KL. The per-trajectory soft loss is

$$\mathcal{L}^R_{\mathrm{soft}}(x, \mathcal{R}) = \frac{1}{L} \sum_{t=1}^{L} D_{\mathrm{KL}}\left(S^{(t)}_{\mathrm{soft}} \| T^{(t)}_{\mathrm{soft}} \right).$$

The local posterior correction becomes

$$\begin{aligned} \widetilde{\mathcal{L}}^R_{\text{post-local}}(t) &= \sum_{u \in \mathcal{V}} S^{(t)}_{\mathrm{mix,local}}(u) D_{\mathrm{KL}}\left(P_S^{(t)}(\cdot \mid u) \| P_T^{(t)}(\cdot \mid u) \right) \\ &= \sum_{u \in \mathcal{V}} \sum_{v \in \mathcal{V}_b^k} q_b(v) S_v^{(t)}(u) \log \frac{P_S^{(t)}(v \mid u)}{P_T^{(t)}(v \mid u)}. \end{aligned}$$

Thus, the outer weighting distribution is the **student local mixture**, rather than the teacher local mixture used for forward KL. Gradients propagate through these student probabilities as well as the student log posterior. The total objective uses the same trajectory-length normalization and the same unit coefficient for the correction, replacing the forward-KL terms in Section 4.3 with their reverse-KL counterparts.

C Experimental Details

C.1 Training Setup

Models and data. The teacher checkpoint is `Qwen/Qwen3-8B`, and the student checkpoints are `Qwen/Qwen3-1.7B-Base` and `Qwen/Qwen3-4B-Base`. We use the non-thinking prompt setting. For the 1.7B experiment, we sample 4,096 prompts from the `all` configuration and `train` split of `open-r1/DAPO-Math-17k-Processed`, with shuffling enabled. Thus, this experiment uses a subset rather than the full DAPO-Math-17k dataset.

Optimization and adaptation. Table 3 summarizes the training configuration. The student is adapted with LoRA (Hu et al., 2022), applied to the attention projections (`q_proj`, `k_proj`, `v_proj`, and `o_proj`) and feed-forward projections (`gate_proj`, `up_proj`, and `down_proj`). The effective prompt batch size is 32, obtained from a micro-batch size of 16 and two gradient accumulation steps. We generate one on-policy trajectory per training prompt, with a maximum of 2,048 new tokens and termination at EOS. Hard-token rollout steps use temperature 1.0 and top-p 0.8.

Table 3: Training hyperparameters for the BASED experiment.

Hyperparameter	Value
Training prompts	4,096
Training epochs / optimization steps	1 / 128
Effective prompt batch size	32
Learning rate	5×10^{-6}
Weight decay	0
Maximum gradient norm	1.0
Numerical precision	`bfloat16`
LoRA rank / scaling parameter	64 / 64
LoRA dropout	0
Maximum prompt length	2,048 tokens
Maximum training completion length	2,048 tokens
Rollouts per prompt	1
Rollout temperature / top-p	1.0 / 0.8
Random seed	42

C.2 BASED Configuration

For the main experiment, BASED uses forward KL with distillation temperature 1.0. Soft trajectories use the top three tokens, a normalized top-k entropy threshold of 0.7, and mixture temperature $\beta = 1.0$. Non-anchor positions follow the hard-token rollout sampling distribution. For posterior correction, each candidate target is the immediate next-token prediction after a soft anchor. We select up to $N_{\text{post}} = 50$ targets per trajectory by highest target entropy. The correction is local: it resolves only the immediately preceding soft anchor. Posterior supervision uses the full aligned vocabulary. The correction has unit weight and is normalized by the number of all trajectory tokens, rather than the number of selected targets. Gradients through cached prefixes are computed using exact cache vector–Jacobian products (`exact_cache_vjp_seq`).

C.3 Evaluation Details

Benchmark sources. Table 4 lists the dataset repositories, splits, and sampling budgets. We evaluate all examples in each specified split. The `train` split names for AIME 2024 and HMMT February 2026 follow the source repositories; these splits are used for evaluation, not for student training.

Table 4: Evaluation datasets and the number n of sampled responses per problem. Repository identifiers refer to Hugging Face datasets.

Benchmark	Repository	Split	n
MATH-500	`HuggingFaceH4/MATH-500`	test	8
AIME 2024	`HuggingFaceH4/aime_2024`	train	32
AIME 2025	`math-ai/aime25`	test	32
AIME 2026	`math-ai/aime26`	test	32
HMMT Feb. 2026	`MathArena/hmmt_feb_2026`	train	32
AMC 2023	`math-ai/amc23`	test	32
MinervaMath	`math-ai/minervamath`	test	8
GPQA-D	`fingertap/GPQA-Diamond`	test	8

Decoding and prompts. Evaluation uses standard discrete-token generation with vLLM (Kwon et al., 2023), temperature 1.0, and top-p 0.8. The maximum prompt length is 2,048 tokens, and the maximum completion length is 8,192 tokens, with termination at EOS. The base random seed is 42. The non-thinking prompt setting is used throughout. For mathematical problems, the system prompt is:

```
Please reason step by step, and put your final answer within
\boxed{}.
```

For GPQA-D, we preserve the answer-option order in the source dataset and use:

```
Please reason step by step and select the single correct
option.  Put only its letter (A, B, C, or D) within \boxed{}
as your final answer.
```

Answer verification and metrics. Mathematical answers are evaluated with Math-Verify,[1] with its precision parameter set to six. GPQA-D uses multiple-choice answer checking against the reference option letter. For a benchmark $\mathcal{D}$, let c_i be the number of correct responses among the n samples for problem i. We compute Pass@k using the standard estimator (Chen et al., 2021):

$$\widehat{\text{Pass@}k}(\mathcal{D}) = \frac{1}{|\mathcal{D}|}\sum_{i=1}^{|\mathcal{D}|}\left[1 - \frac{\binom{n-c_i}{k}}{\binom{n}{k}}\right], \qquad k \in \{1, 8\}, \tag{1}$$

where $\binom{n-c_i}{k} = 0$ when $n - c_i < k$. Pass@1 is therefore the mean accuracy over sampled responses, not greedy-decoding accuracy. For $n = 8$, Pass@8 is the fraction of problems with at least one correct response; for $n = 32$, all 32 samples contribute to estimating Pass@8, rather than Pass@32. Scores are reported as percentages. AVG is computed by averaging the eight benchmark-level scores with equal weights. All averages and performance differences are computed before rounding.

D Additional Details of Trajectory Diversity Analysis

Data and sampling. We analyze the final checkpoints of vanilla OPD, soft only, and BASED on two AIME 2026 problems. Q4 and Q15 denote zero-based indices in the dataset returned by our evaluation loader. For each problem and method, we sample 512 responses using shared prompt and decoding settings. Responses are partitioned into correct and incorrect subsets using our `math_verify`-based evaluator.

Self-BLEU. For each problem and method, we randomly select four correct responses without replacement within each repetition. Following Zhu et al. (2018), we treat each selected response as the hypothesis and the other three as references, compute its sentence-level cumulative BLEU-4, and average the four scores. We repeat this procedure 50 times and report the mean on a 0–100 scale. We use case-sensitive WordPunct tokenization and method-1 smoothing with $\epsilon = 0.1$ (Chen & Cherry, 2014). The subset size is fixed across methods so that each hypothesis is evaluated against the same number of references.

chrF-based trajectory clustering. We adopt the trajectory-level clustering procedure of Matsutani et al. (2026). For each problem and method, we cluster all correct responses and all incorrect responses separately, without balancing the sizes of these subsets. Pairwise similarity is measured using chrF (Popović, 2015), with $\beta_{\text{chrF}} = 2$, character n-grams up to order six, and no word n-grams. Scoring is case-sensitive and excludes whitespace. We average the two directional chrF scores and divide by 100 to obtain a symmetric similarity $s_{ij} \in [0, 1]$, then use $d_{ij} = 1 - s_{ij}$ for UPGMA hierarchical clustering with average linkage. We cut each dendrogram at distance 0.45, corresponding to a similarity threshold of 55 on the 0–100 scale, and report the resulting numbers of correct and incorrect clusters. The same threshold is used for every problem and method.

[1] https://github.com/huggingface/Math-Verify

E Additional Results with Reverse KL

Table 5 reports reverse-KL (RKL) results for Qwen3-1.7B-Base. The reverse-KL formulation of BASED is given in Appendix B.3.

Table 5: Results (%) under reverse KL with Qwen3-8B in non-thinking mode as the teacher. AVG is the unweighted average over the eight benchmarks. Bold indicates the best result within each metric block, including ties.

Method	MATH 500	AIME 2024	AIME 2025	AIME 2026	HMMT Feb. 26	AMC 2023	Minerva Math	GPQA-D	**AVG**
Student: Qwen3-1.7B-Base									
					Pass@1				
Base	17.48	1.15	1.25	0.63	0.38	10.23	8.59	6.50	5.78
+ OPD	64.90	7.81	**5.52**	**4.69**	1.42	34.77	27.99	25.06	21.52
+ **BASED**	**65.45**	**8.85**	4.69	4.48	**2.08**	**36.02**	**28.58**	**25.88**	**22.00**
					Pass@8				
Base	62.00	7.81	8.72	4.81	2.13	43.44	34.56	37.88	25.17
+ OPD	85.00	19.41	**19.61**	**16.59**	**6.78**	**70.59**	**58.09**	**78.79**	**44.36**
+ **BASED**	**87.20**	**22.99**	14.95	12.72	6.49	67.38	52.57	76.77	42.63

Compared with vanilla OPD under RKL, BASED slightly improves average Pass@1 (21.52 to 22.00) but reduces average Pass@8 (44.36 to 42.63). Relative to forward-KL BASED in Table 1, RKL also yields higher average Pass@1 (22.00 vs. 20.57) and slightly lower average Pass@8 (42.63 vs. 42.79). We hypothesize that this trade-off is related to RKL's mode-seeking tendency, which may encourage the student to concentrate on a dominant reasoning trajectory rather than preserve broader coverage of alternatives. This could favor single-sample accuracy while limiting the benefit of multiple samples.