LBSD: LOCAL-BRANCH SELECTIVE SELF-DISTILLATION

Anonymous authors
Paper under double-blind review

ABSTRACT

On-policy self-distillation (OPSD) mitigates the coarse-grained supervision issue of reinforcement learning with verifiable rewards (RLVR) with long reasoning trajectories by introducing dense token-level targets. However, standard OPSD methods largely treat all tokens alike, overlooking that only a subset of positions provides strong signals for exploration or correction. We introduce LBSD, a **L**ocal-**B**ranch **S**elective self-**D**istillation framework that concentrates supervision on informative parts of the reasoning process. LBSD extends single-trajectory OPSD by selectively expanding critical intermediate states into local on-policy branches. It uses student entropy and teacher-student discrepancy to identify informative branches and token positions for learning. This design provides broader and more targeted feedback. Moreover, we propose abstract solution strategies as privileged teacher information instead of full reference reasoning trajectories, reducing the risk of answer leakage. Experiments across four competition mathematics benchmarks and multiple Qwen3 model scales demonstrate consistent improvements over representative RLVR and self-distillation baselines. Ablation studies further support the effectiveness of local branching and distribution-informed selection. These findings suggest that effective self-distillation depends not only on providing dense targets, but also on identifying where and along which local continuations those targets should be applied.

1 INTRODUCTION

Reinforcement learning with verifiable rewards (RLVR) has become a widely used paradigm for improving the reasoning ability of large language models (LLMs) on multistep tasks such as mathematics and code generation. Methods such as GRPO and DAPO optimize policies with outcome rewards, enabling models to improve through exploration guided by task outcomes (Shao et al., 2024; Yu et al., 2025; DeepSeek-AI, 2025). However, outcome-only rewards provide sparse supervision over long reasoning trajectories, indicating whether a sampled trajectory is ultimately correct without directly assigning credit to the intermediate reasoning steps that determine the outcome. On-policy distillation (OPD) addresses this limitation by training the student on trajectories generated by its current policy while querying a teacher for dense next-token distributions at each visited prefix (Agarwal et al., 2024; Li et al., 2026b). Because supervision is provided on student-generated trajectories, the teacher can directly guide the states and errors encountered by the evolving policy. On-policy self-distillation (OPSD) further extends this formulation to a self-teacher setting, where the same model serves as student and teacher under different contexts (Zhao et al., 2026a). The student reasons from the problem statement alone, while the teacher conditions on privileged solution information, providing dense supervision without a separately trained external teacher.

Despite providing dense token-level feedback, existing OPSD still faces limitations in the coverage and allocation of its supervision, as well as in the privileged information. First, standard OPSD collects supervision only along trajectories sampled from the current student policy. Each rollout supplies distillation targets only along its realized trajectory, while other possible trajectories are not represented in the supervision. This restricted supervision support provides only a partial view of the student's possible reasoning behavior, particularly in high-entropy regions. Second, dense distillation supervision is not equally informative across token positions. Recent studies suggest that the effectiveness of on-policy distillation targets can vary substantially across different regions of the student's reasoning process (Zheng et al., 2026; Xu et al., 2026). Since some regions expose

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

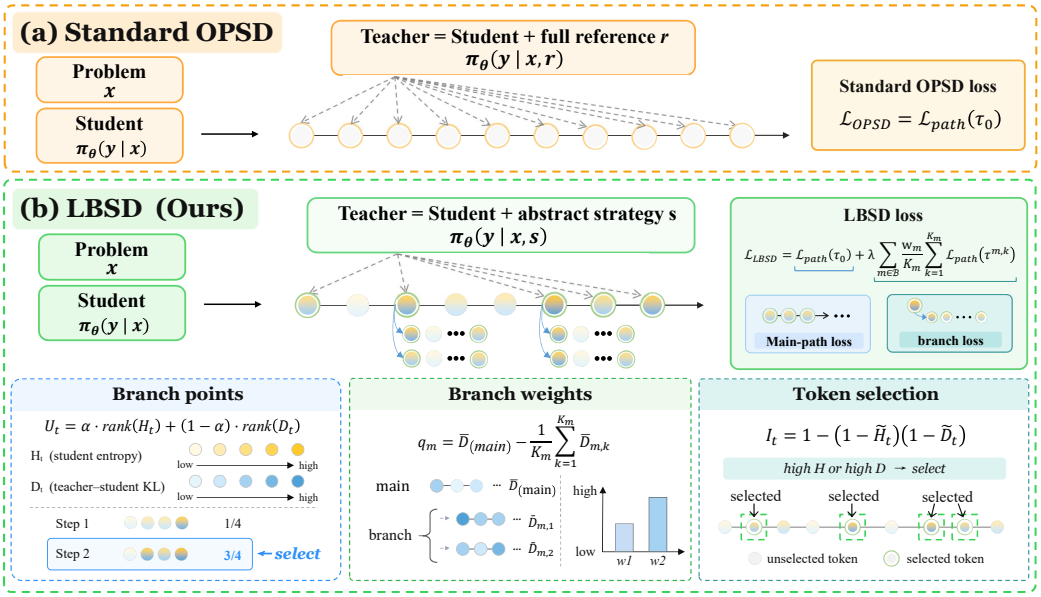


Figure 1: Overview of standard OPSD and LBSD. Standard OPSD distills a single student trajectory, while LBSD expands informative intermediate states with local branch and selectively distills useful branches and tokens, using an abstract solution strategy as privileged teacher information.

strong uncertainty or teacher–student discrepancy, these tokens can provide substantial opportunities for exploration or correction. However, assigning equal weight to all token-level supervision allocate substantial optimization effort to positions that provide limited additional learning signal. Instead, it is intuitive that directing supervision toward more informative learning signals enables more effective self-distillation. Finally, existing OPSD typically conditions the privileged teacher on a full reference solution. The reference contains the target answer and detailed derivations that are unavailable to the student. As a result, the teacher's predictions may reflect reference-specific information in addition to broadly transferable reasoning guidance. Distilling such predictions may therefore introduce answer leakage or reference-induced shortcuts, encouraging the student to rely on information that will not be available at inference time.

To address these limitations, we propose **L**ocal **B**ranch **S**elective Self **D**istillation (LBSD). As illustrated in Figure 1, LBSD broadens on-policy supervision through local branching and uses abstract problem solving strategies as privileged teacher context. First, LBSD selectively branches from informative intermediate states by sampling multiple local on-policy continuations from each selected state. This targeted expansion broadens the on-policy supervision available for each training problem beyond the single trajectory sampled by standard OPSD. Second, LBSD selectively prioritizes the expanded supervision at both branch and token levels. At the branch level, LBSD estimates branch informativeness by comparing the teacher–student discrepancy of local continuations with that of the original continuation. At the token level, LBSD selects informative positions using student entropy and teacher–student discrepancy as complementary signals of exploration and correction. Together, these two levels direct distillation toward informative branches and positions rather than treating all expanded supervision equally. Finally, LBSD replaces the full reference solution in the privileged teacher context with abstract problem solving strategies. These strategies retain transferable problem solving guidance while withholding explicit answer derivations and other reference specific details that may induce privileged information leakage or reference specific shortcuts, thereby reducing the influence of answer specific reference information on its supervision.

Experiments across four competition mathematics benchmarks and three Qwen3 model scales show that LBSD achieves the highest macro average at every evaluated scale. Further analyses show that local branching exposes additional teacher–student discrepancy around selected branch points,

while using abstract strategies as privileged context consistently reduces privileged context leakage compared with standard OPSD using full reference solutions.

In summary, our contributions are as follows:

• We introduce LBSD, which extends on-policy self-distillation beyond a single student trajectory by selectively expanding informative intermediate states with local on-policy continuations and concentrating supervision on useful branches and token positions.

• We replace full reference solutions in the privileged context with abstract solution strategies, retaining high-level guidance while reducing answer leakage and reference-dependent supervision.

• Experiments across four competition mathematics benchmarks and three Qwen3 model scales consistently improve over RLVR and self-distillation baselines, while controlled analyses validate the benefits of local branching, selective supervision, and abstract privileged strategies.

2 Related Work

Reinforcement Learning with Verifiable Rewards Reinforcement learning with verifiable rewards (RLVR) improves reasoning models by optimizing policies with automatically checkable signals, such as math answer verification or code execution results. DeepSeekMath, DeepSeek-R1, and DAPO demonstrate the effectiveness of GRPO-style or rule-based RL for mathematical and long chain reasoning (Shao et al., 2024; DeepSeek-AI, 2025; Yu et al., 2025). Recent analyses further highlight the roles of exploration coverage, token-level updates, and high-entropy positions in RLVR, suggesting that reasoning improvement depends not only on outcome rewards but also on where learning signals are applied (Yang et al., 2025; Huang et al., 2026; Wang et al., 2025b). These findings motivate fine-grained supervision that targets informative states rather than propagating outcome-level signals uniformly across long reasoning trajectories.

Search-based and Branch-structured Reasoning Search-based reasoning explores alternative continuations through sampling, tree, or graph structures (Wei et al., 2022; Wang et al., 2023; Yao et al., 2023; Besta et al., 2024; Hao et al., 2023). Recent work further incorporates branch structure into post-training, using reasoning trees to broaden RL exploration or schedule training examples according to trajectory structure (Hou et al., 2025; Wang et al., 2025a). These studies demonstrate that intermediate reasoning states and their alternative continuations provide information beyond a single realized path. Unlike prior work that primarily uses branches for inference-time search, solution selection, or outcome-based exploration, LBSD constructs a small set of local on-policy branches specifically as additional support for self-distillation and selectively weights their supervision.

On-Policy Self-Distillation for Reasoning On-policy distillation (OPD) trains on trajectories generated by the current student policy and queries a teacher for dense token-level supervision, reducing the distribution mismatch caused by fixed teacher-generated responses (Agarwal et al., 2024). On-policy self-distillation (OPSD) further instantiates the teacher and student from the same model under different contexts, allowing privileged information to supervise student rollouts without requiring a separate teacher model (Zhao et al., 2026a). Subsequent work extends OPSD along two main directions. One direction combines self-distillation with reward-based policy optimization, assigning complementary roles to verifiable rewards and distributional supervision through token-level update modulation or sample routing (Yang et al., 2026; Li et al., 2026a). Another direction improves the reliability of supervision along the original student rollout by reweighting or localizing distillation according to teacher uncertainty, sequence position, or detected reasoning errors (Ke et al., 2026; Liu et al., 2026; Zhao et al., 2026b). In contrast, LBSD augments the original rollout with local on-policy continuations from informative intermediate states, extending self-distillation beyond a single realized trajectory.

3 Methods

We propose LBSD, a local-branch selective self-distillation method for mathematical reasoning. Following OPSD, the model generates reasoning trajectories under a student context and learns

from a stop-gradient teacher distribution evaluated on the same prefixes. The student observes only the problem, whereas the teacher additionally receives an abstract solution strategy.

LBSD extends the original trajectory with several local branches sampled from selected intermediate states. It uses student uncertainty and teacher–student discrepancy to identify informative states, weight the resulting branches, and concentrate distribution distillation on useful token positions.

3.1 Privileged Strategy Construction

The teacher in LBSD is conditioned on an abstract solution strategy rather than a complete reference reasoning trajectory. For each training example, we derive the strategy from the problem and an available reference solution. The strategy retains only high-level information, including the problem type, core intuition, strategic direction, and potential pitfalls. The extraction prompt excludes information that would directly reveal or reproduce the reference solution. It instructs the model to omit the final answer, complete derivations, copied equations or procedures, decisive intermediate values, and final case conclusions. The complete prompt is provided in Appendix B.2.

The resulting strategy provides the teacher with a high-level reasoning plan while withholding the answer-bearing derivation. The teacher must reconstruct the intermediate reasoning and perform the necessary calculations when evaluating prefixes generated by the student. This construction reduces direct answer leakage and limits imitation of specific expressions, intermediate values, and surface forms. At the same time, the strategy supplies information unavailable to the student, enabling the teacher to provide informed token-level guidance without changing the inference-time context.

3.2 Informative Local Branching

Given a problem x, the student first samples an on-policy reasoning trajectory $\tau^0 = (y_1, \ldots, y_T)$ under the student context $C_s(x)$, which we refer to as the main path. We segment the trajectory into reasoning steps using natural textual boundaries such as newlines and punctuation, while imposing a maximum step length to avoid overly coarse segments. These steps define candidate locations from which alternative local continuations can be explored.

Rather than branching uniformly along the main path, LBSD selects informative steps using signals that characterize both exploration and correction. At each token position t, let H_t denote the entropy of the student's next-token predictive distribution, and let D_t denote the teacher–student discrepancy measured by forward KL divergence:

$$D_t = D_{\mathrm{KL}}\left(p_\theta^T(\cdot \mid C_T(x, s^*), \tau^0_{<t}) \,\middle\|\, p_\theta^S(\cdot \mid C_s(x), \tau^0_{<t})\right). \tag{1}$$

Student entropy serves as an exploration signal, while teacher–student discrepancy serves as a signal of potential correction. We apply rank normalization to the two signals and combine them as

$$U_t = \alpha \cdot \mathrm{rank}(H_t) + (1 - \alpha) \cdot \mathrm{rank}(D_t), \tag{2}$$

where α controls their relative contribution. For each reasoning step, steps containing more token positions above the predefined quantile threshold receive higher scores. Among the candidate steps satisfying the length and position constraints, we select those with the highest scores as branch points. This procedure concentrates branch expansion on intermediate states that exhibit stronger exploration and correction signals. For each selected branch point $m \in \mathcal{B}$, LBSD samples K_m local on-policy continuations:

$$\tau^{m,k}_{\geq m} \sim p_\theta^S\left(\cdot \mid C_s(x), \tau^0_{<m}\right), \qquad k = 1, \ldots, K_m. \tag{3}$$

The resulting training structure consists of the original main path together with several groups of local branch paths. Local branching provides multiple realizations of how the student may continue reasoning from the same selected point. By expanding states with stronger exploration or correction signals, the sampled continuations are more likely to expose meaningful variation in the student's reasoning behavior, while extending the coverage of on-policy supervision beyond the single continuation observed on the main path.

3.3 Selective Distillation at Branch and Token Levels

The local branching procedure expands the set of on-policy continuations, but the resulting supervision need not contribute equally to distillation. LBSD therefore selectively allocates supervision at

two levels: it first estimates the alignment of branching from each selected state, and then identifies informative token positions within the resulting continuations.

At the branch level, we evaluate whether the local continuations generated from a branch point provide supervision that is better aligned with the privileged teacher than the original continuation. For a selected branch point m, let $\bar{D}_{\text{main}}(m)$ denote the average teacher–student discrepancy over a fixed window following m on the main path, and let $\bar{D}_{m,k}$ denote the corresponding discrepancy on the k-th local continuation. We define the branch alignment as

$$q_m = \bar{D}_{\text{main}}(m) - \frac{1}{K_m} \sum_{k=1}^{K_m} \bar{D}_{m,k}. \tag{4}$$

A larger q_m indicates that the local continuations reduce teacher–student discrepancy more substantially relative to the main continuation. We apply a softmax over the branch alignment q_m across the selected branch points to obtain normalized weights w_m. Branch groups with larger estimated alignment therefore receive greater weight in the final distillation objective.

Within each path, LBSD further selects informative token positions using student entropy and teacher–student discrepancy as complementary signals of exploration and correction. For each effective training position t, we compute the normalized student entropy $\tilde{H}_t$ and teacher–student discrepancy $\tilde{D}_t$, and define the token importance score as

$$I_t = 1 - (1 - \tilde{H}_t)(1 - \tilde{D}_t). \tag{5}$$

The score increases when either the student exhibits a stronger exploration signal or the privileged teacher provides a stronger corrective signal. We retain the top fraction of positions according to I_t and denote the resulting token set for path τ by $\Omega(\tau)$. Given the selected positions, the distillation loss for a path τ is

$$\mathcal{L}_{\text{path}}(\tau) = \frac{1}{|\Omega(\tau)|} \sum_{t \in \Omega(\tau)} D_{\text{KL}} \left(\text{sg}\left[p_\theta^T(\cdot \mid C_T(x, s^*), \tau_{<t}) \right] \,\middle\|\, p_\theta^S(\cdot \mid C_s(x), \tau_{<t}) \right). \tag{6}$$

The effective training region depends on the path type. For the main path, supervision is applied to the generated completion. For a local branch, the loss is computed only over the newly sampled suffix after the branch point. This avoids repeatedly distilling identical prefix tokens and confines branch supervision to the region where the alternative continuation begins.

Finally, LBSD combines the main-path loss with the weighted local-branch losses:

$$\mathcal{L}_{\text{distillation}} = \mathcal{L}_{\text{main}} + \lambda \sum_{m \in \mathcal{B}} w_m \frac{1}{K_m} \sum_{k=1}^{K_m} \mathcal{L}_{m,k}. \tag{7}$$

Here, $\mathcal{L}_{\text{main}}$ denotes the distillation loss on the main trajectory, $\mathcal{L}_{m,k}$ denotes the loss on the k-th local continuation sampled from branch point m, and λ controls the overall contribution of local-branch supervision relative to the main path. Together, branch-level weighting and token-level selection allow LBSD to use the expanded on-policy supervision selectively. Branch weighting emphasizes locations whose alternative continuations achieve better alignment with the privileged teacher relative to the original continuation, while token selection concentrates distillation on positions carrying stronger exploration or correction signals.

4 Experiments

4.1 Experimental Setup

Models and datasets. We use Qwen3-1.7B, Qwen3-4B, and Qwen3-8B as base models (Qwen-Team, 2025). Training examples are drawn from DeepMath (He et al., 2025). We retain only examples with difficulty greater than 7.0. We evaluate on competition-level mathematics benchmarks including AIME25, AIME26, HMMT25, and HMMT26 (Dekoninck et al., 2026). For each benchmark problem, we independently sample 36 responses and report their mean correctness as Avg@36.

Table 1: Avg@36 performance (%) on competition mathematics benchmarks. The best and second-best results within each model scale are **bolded** and underlined.

		AIME		HMMT		
Method	**Base Model**	2025	2026	2025	2026	**Avg.**
GRPO (Shao et al., 2024)	Qwen3-1.7B	35.37	38.52	23.33	28.28	31.38
OPSD (Zhao et al., 2026a)	Qwen3-1.7B	38.15	40.83	**25.37**	28.79	33.28
RLSD (Yang et al., 2026)	Qwen3-1.7B	36.94	41.11	24.35	29.46	32.97
RLCSD (Pan et al., 2026)	Qwen3-1.7B	35.37	38.70	23.70	27.95	31.43
LBSD (Ours)	Qwen3-1.7B	**39.91**	**44.17**	25.00	**29.55**	**34.66**
GRPO (Shao et al., 2024)	Qwen3-4B	64.35	66.02	43.98	40.91	53.81
OPSD (Zhao et al., 2026a)	Qwen3-4B	67.22	65.18	42.96	41.41	54.20
RLSD (Yang et al., 2026)	Qwen3-4B	65.74	65.93	42.31	39.98	53.49
RLCSD (Pan et al., 2026)	Qwen3-4B	66.11	65.18	43.61	41.50	54.10
LBSD (Ours)	Qwen3-4B	**69.26**	**67.41**	**44.08**	**41.83**	**55.37**
GRPO (Shao et al., 2024)	Qwen3-8B	68.89	65.83	43.17	45.13	55.75
OPSD (Zhao et al., 2026a)	Qwen3-8B	70.09	68.33	44.54	45.03	57.00
RLSD (Yang et al., 2026)	Qwen3-8B	68.33	67.78	43.89	**46.13**	56.53
RLCSD (Pan et al., 2026)	Qwen3-8B	69.35	68.33	45.09	44.78	56.89
LBSD (Ours)	Qwen3-8B	**71.85**	**70.93**	**46.30**	45.79	**58.72**

Baselines. We compare LBSD with four representative post-training baselines. GRPO (Shao et al., 2024) represents an RLVR objective that learns from verifiable outcome rewards. OPSD (Zhao et al., 2026a) performs on-policy distillation by supervising student-generated trajectories with a stop-gradient teacher distribution. RLSD (Yang et al., 2026) is a self-distillation RLVR baseline that converts rollout feedback into a denser learning signal. RLCSD (Pan et al., 2026) contrasts teacher–student gaps under correct and incorrect hints to suppress style drift and focus supervision on task-relevant tokens.

LBSD configuration. LBSD selects three branch points from each main trajectory and samples two local continuations from every selected point. The branch score combines student entropy and teacher–student discrepancy with equal weights by setting $\alpha = 0.5$. We retain 80% of the tokens according to the token importance score. We use Qwen3.5-27B (Qwen Team, 2026) only for offline extraction of a high-level solution strategy from the first reference solution of each DeepMath example. Full training and evaluation configurations are provided in Appendix C.

4.2 Main Results

Table 1 presents the main results on competition mathematics benchmarks. LBSD consistently delivers the strongest overall reasoning performance across all three model scales, outperforming the best competing method by 1.38, 1.17, and 1.72 points on average with Qwen3-1.7B, Qwen3-4B, and Qwen3-8B. This advantage is highly consistent: LBSD ranks first in 10 of the 12 model–benchmark settings. Notably, LBSD achieves the best performance on every AIME setting, with improvements of up to 3.06 points over the RLSD. Overall, these results demonstrate that LBSD provides consistent state-of-the-art performance compared with other strong baselines on several competition benchmarks.

4.3 Analysis of Local Branch Supervision

In this section, we investigate what additional supervision local branching provides beyond selecting informative tokens on the realized main trajectory. We sample problems from the DeepMath training set, follow the training-time branching procedure, and compare teacher–student discrepancy between each branch and its matched main continuation over up to 200 post-branch tokens. Inspection of the selected steps shows that the entropy–discrepancy score frequently identifies decision points in the reasoning process, where regenerating the selected step can change the direction of the later continuation. Figure 2(a) shows a consistent non-monotonic pattern in the branch–main dis-

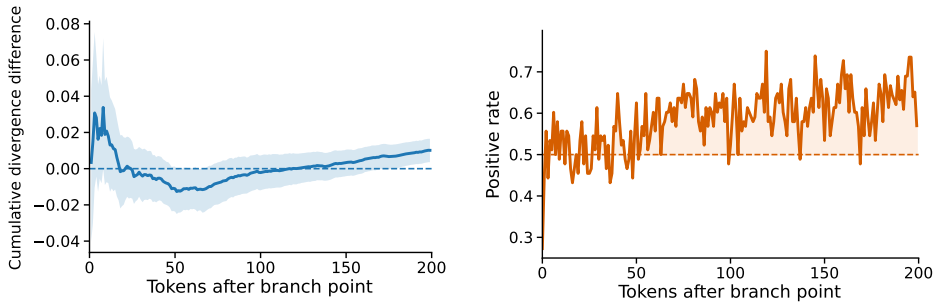


Figure 2: Dynamics of teacher–student discrepancy on local branches relative to matched main continuations. Left: Cumulative mean branch–main discrepancy difference over aligned post-branch positions. Right: Fraction of problems in which the local branch has higher discrepancy than the matched main continuation at each post-branch position.

crepancy difference, with a clear recovery at later positions after an intermediate drop. Figure 2(b) further shows that positions where branch discrepancy exceeds the matched main continuation become more common later in the rollout.

For branches with increased downstream discrepancy, we find that regenerating the selected step often changes the local reasoning decision, leading the continuation to follow a different reasoning direction or rely on a different intermediate claim. The resulting teacher–student disagreement becomes more visible as the alternative continuation develops. Taken together, these results show that local branching does more than expand on-policy supervision coverage: it changes the local teacher–student discrepancy distribution and enriches the expanded supervision pool with informative discrepancy-bearing positions that are absent from the single realized continuation. This gives subsequent token-level selection a larger and more informative set of candidate positions, with more positions carrying strong correction signals.

4.4 Analysis of Privileged Context Leakage

We investigate how the form of privileged teacher information affects student behavior beyond task accuracy by comparing Base model, OPSD, and LBSD on DeepMath and AIME26. DeepMath is used for post-training, whereas AIME26 is used only for evaluation, allowing us to examine whether reference-induced behaviors are confined to the training data or persist beyond it. If full reference trajectories introduce reference-specific supervision through dense distillation, this influence should appear not only as explicit references to unavailable privileged information but also as broader shifts toward answer- and solution-oriented generation patterns. We therefore measure Leakage Rate to-

Table 2: Privileged-context leakage and response-style deviations of Qwen3-4B on DeepMath and AIME26. Response-style columns report absolute deviations from the corresponding Base model; lower is better. Green marks the smaller deviation between OPSD and LBSD.

	Leakage	**Response Style Deviation** ($\lvert\Delta_{\text{Base}}\rvert \downarrow$)			
Method	Rate (%) ↓	Answer Density	Solution Density	Boxed Freq.	Final-Answer Freq.
DeepMath					
Qwen3-4B	1.00	0.0000	0.0000	0.0000	0.0000
OPSD	1.25	0.9835	0.1722	0.5350	0.0941
LBSD	0.75	0.5654	0.0318	0.4183	0.1533
AIME26					
Qwen3-4B	0.92	0.0000	0.0000	0.0000	0.0000
OPSD	1.25	0.3177	0.1775	0.1358	0.0550
LBSD	0.67	0.0118	0.0286	0.0075	0.0050

gether with response-style statistics that characterize such generation patterns. For this analysis, we collect rollouts from DeepMath and AIME26 using the same 1,024-token generation limit as in training. Leakage Rate is obtained by manually identifying responses that reveal privileged information unavailable to the student. The response-style statistics capture broader behavioral shifts by measuring the frequency of predefined answer- and solution-related expressions in the same rollouts. Detailed metric definitions and annotation criteria are provided in Appendix C.

As shown in Table 2, OPSD exhibits consistently stronger privileged-context leakage than LBSD on both the post-training and held-out data, suggesting that the influence of full-reference supervision is not confined to the training examples. The response-style statistics reveal a broader behavioral effect: OPSD tends to shift generation away from the base model toward more answer- and solution-oriented patterns, while LBSD largely preserves the original response behavior, with the distinction being particularly clear on the held-out AIME26 benchmark. Although this tendency is not uniform across every statistic on DeepMath, the overall pattern indicates that full reference trajectories can leave a reference-specific imprint on the student beyond explicit leakage alone. These observations are consistent with a mechanism in which dense distillation propagates reference-specific generation patterns from the privileged teacher context, whereas abstract solution strategies retain high-level reasoning guidance while reducing the influence of answer-bearing and trajectory-specific details.

4.5 Ablation Studies

We conduct controlled ablations on Qwen3-4B to isolate the contributions of privileged teacher information, local branching, branch-point selection, and branch weighting. As shown in Table 3, all ablations reduce the average score from 56.67 to between 55.60 and 56.07, indicating that the LBSD's improvement does not arise from a single component.

Table 3: Ablation of LBSD components on Qwen3-4B. "Full Ref. Sol." denotes the full reference solution

Variant	AIME25	HMMT25	Avg.
Full Ref. Sol.	68.61	42.59	55.60
w/o Branch	68.33	<u>43.52</u>	55.93
Random Branch	<u>68.80</u>	43.33	<u>56.07</u>
w/o Branch Weight	68.06	43.24	55.65
LBSD (Ours)	**69.26**	**44.08**	**56.67**

The Full Ref. Sol. variant replaces the abstract strategy with the full reference solution; together with the privileged-context analysis in Section 4.4, its lower performance suggests that more detailed privileged information does not necessarily provide more effective supervision. The w/o Branch variant restricts distillation to the main trajectory and lowers the average score to 55.93. Random Branch introduces local continuations from randomly selected branch points and raises the score to 56.07, but remains 0.60 points below full LBSD. This contrast shows that additional on-policy continuations are useful, but their benefit depends on expanding informative intermediate states rather than allocating the branch budget arbitrarily. Finally, w/o Branch Weight removes branch-level weighting and reduces the average score to 55.65, a 1.02-point drop from LBSD, showing that expanded branch supervision should not be treated uniformly. Overall, these results support the two central principles of LBSD: broadening supervision beyond the single realized trajectory and selectively concentrating the expanded supervision on informative branches.

4.6 Parameter Analysis

Branch-budget allocation. We examine whether LBSD is sensitive to how a fixed local-continuation budget is distributed between the number of branch points B and the number of continuations per point K. With $K \times B = 6$, the four tested allocations achieve AIME25 scores between 68.52 and 69.26, with a total variation of only 0.74 points. Performance also shows no monotonic trend as the budget shifts from fewer branch points with more continuations to more branch points with fewer continuations. These results indicate that the benefit of local branching is not tied to a particular allocation of the continuation budget and remains stable across the tested configurations.

Mixing-coefficient sensitivity. We vary the mixing coefficient α that controls the relative contribution of student entropy and teacher–student discrepancy in branch-point scoring. With $\alpha \in \{0.3, 0.5, 0.7\}$, AIME25 scores are 68.52, 69.26, and 68.80, respectively. Equal weighting achieves

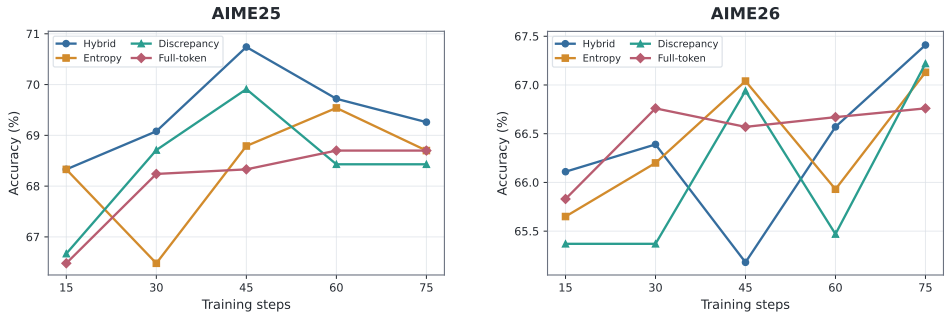


Figure 3: Training dynamics of different token selection criteria on AIME25 and AIME26. The selective variants retain 80% of token positions. Each point denotes the mean Avg@36 at the corresponding training step, while the full-token curve reports results without token filtering.

the highest observed performance, while shifting the weight toward either entropy or discrepancy does not provide further improvement. These results support combining the two signals without strongly favoring either one in branch-point selection.

Token selection. We compare entropy-based, discrepancy-based, and hybrid token selection under the same token keep ratio of $r = 0.8$, using full-token distillation as a reference. At the final checkpoint, hybrid selection reaches 69.26 on AIME25 and 67.41 on AIME26, outperforming full-token distillation at 68.70 and 66.76, respectively. This consistent improvement over full-token supervision shows that selectively concentrating distillation on informative positions can yield better final performance than supervising all tokens uniformly. The advantage is also persistent during training: hybrid selection achieves the best or tied-best performance at 9 of the 10 reported checkpoints across the two benchmarks. This pattern indicates that the benefit of the hybrid criterion is not confined to a single training stage. Meanwhile, the relative performance of entropy-only and discrepancy-only selection varies across checkpoints and benchmarks, with neither signal consistently dominating the other. Their changing relative effectiveness, together with the stable advantage of the hybrid criterion, supports treating student entropy and teacher–student discrepancy as complementary signals for identifying informative distillation positions.

Table 4: Sensitivity of LBSD configurations on AIME25.

Configuration	AIME25
Branch-budget allocation	
1×6	68.52
2×3	**69.26**
3×2	68.52
6×1	68.80
Mixing coefficient α	
0.3	68.52
0.5	**69.26**
0.7	68.80

5 Conclusion

We introduced LBSD, a local-branch selective self-distillation framework for mathematical reasoning that extends on-policy self-distillation beyond a single realized trajectory. LBSD expands informative intermediate states into local on-policy continuations, selectively concentrates supervision at the branch and token levels, and uses abstract solution strategies as privileged teacher context. Across four competition mathematics benchmarks and three Qwen3 model scales, LBSD consistently improves over representative RLVR and self-distillation baselines. Our analyses show that local branching exposes additional positions with stronger teacher–student discrepancy than the main continuation, abstract strategies reduce reference-specific leakage, and entropy and discrepancy provide complementary signals for token selection. Overall, our results highlight three important choices in on-policy self-distillation: which local continuations are explored, which token positions are distilled, and what privileged information is provided to the teacher.

AI USE STATEMENT

Generative AI tools were used to improve the readability and wording of the manuscript. They were also used to assist in checking the consistency between the experimental parameters documented in the appendix and those specified in the code, including identifying potentially omitted documentation. All AI-assisted edits and consistency checks were reviewed and verified by the authors. The authors take full responsibility for the final content of the paper.

REPRODUCIBILITY STATEMENT

We provide an anonymous implementation of LBSD at `https://anonymous.4open.science/r/Local-Branch-Selective-Distillation-AD7C/`.

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 *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL `https://openreview.net/forum?id=3zKtaqxLhW`.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, and Torsten Hoefler. Graph of thoughts: Solving elaborate problems with large language models. In Michael J. Wooldridge, Jennifer G. Dy, and Sriraam Natarajan (eds.), *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2024, February 20-27, 2024, Vancouver, Canada*, pp. 17682–17690. AAAI Press, 2024. doi: 10.1609/AAAI.V38I16.29720. URL `https://doi.org/10.1609/aaai.v38i16.29720`.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *CoRR*, abs/2501.12948, 2025. doi: 10.48550/ARXIV.2501.12948. URL `https://doi.org/10.48550/arXiv.2501.12948`.

Jasper Dekoninck, Nikola Jovanovic, Tim Gehrunger, Kári Rögnvaldsson, Ivo Petrov, Chenhao Sun, and Martin T. Vechev. Beyond benchmarks: Matharena as an evaluation platform for mathematics with llms. *CoRR*, abs/2605.00674, 2026. doi: 10.48550/ARXIV.2605.00674. URL `https://doi.org/10.48550/arXiv.2605.00674`.

Shibo Hao, Yi Gu, Haodi Ma, Joshua Jiahua Hong, Zhen Wang, Daisy Zhe Wang, and Zhiting Hu. Reasoning with language model is planning with world model. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pp. 8154–8173. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.EMNLP-MAIN.507. URL `https://doi.org/10.18653/v1/2023.emnlp-main.507`.

Zhiwei He, Tian Liang, Jiahao Xu, Qiuzhi Liu, Xingyu Chen, Yue Wang, Linfeng Song, Dian Yu, Zhenwen Liang, Wenxuan Wang, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. Deepmath-103k: A large-scale, challenging, decontaminated, and verifiable mathematical dataset for advancing reasoning. *CoRR*, abs/2504.11456, 2025. doi: 10.48550/ARXIV.2504.11456. URL `https://doi.org/10.48550/arXiv.2504.11456`.

Zhenyu Hou, Ziniu Hu, Yujiang Li, Rui Lu, Jie Tang, and Yuxiao Dong. Treerl: LLM reinforcement learning with on-policy tree search. 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), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pp. 12355–12369. Association for Computational Linguistics, 2025. doi: 10.18653/V1/2025.ACL-LONG.604. URL `https://doi.org/10.18653/v1/2025.acl-long.604`.

Kexin Huang, Haoming Meng, Junkang Wu, Jinda Lu, Chiyu Ma, Ziqian Chen, Xue Wang, Bolin Ding, Jiancan Wu, Xiang Wang, Xiangnan He, Guoyin Wang, and Jingren Zhou. On the direction of RLVR updates for LLM reasoning: Identification and exploitation. *CoRR*, abs/2603.22117, 2026. doi: 10.48550/ARXIV.2603.22117. URL https://doi.org/10.48550/arXiv.2603.22117.

Junlong Ke, Zichen Wen, Weijia Li, Conghui He, and Linfeng Zhang. Respecting self-uncertainty in on-policy self-distillation for efficient llm reasoning, 2026. URL https://arxiv.org/abs/2605.13255.

Gengsheng Li, Tianyu Yang, Junfeng Fang, Mingyang Song, Mao Zheng, Haiyun Guo, Dan Zhang, Jinqiao Wang, and Tat-Seng Chua. Unifying group-relative and self-distillation policy optimization via sample routing. *CoRR*, abs/2604.02288, 2026a. doi: 10.48550/ARXIV.2604.02288. URL https://doi.org/10.48550/arXiv.2604.02288.

Yaxuan Li, Yuxin Zuo, Bingxiang He, Jinqian Zhang, Chaojun Xiao, Cheng Qian, Tianyu Yu, Huan-ang Gao, Wenkai Yang, Zhiyuan Liu, and Ning Ding. Rethinking on-policy distillation of large language models: Phenomenology, mechanism, and recipe. *CoRR*, abs/2604.13016, 2026b. doi: 10.48550/ARXIV.2604.13016. URL https://doi.org/10.48550/arXiv.2604.13016.

Xiaogeng Liu, Xinyan Wang, Yingzi Ma, Yechao Zhang, and Chaowei Xiao. When are teacher tokens reliable? position-weighted on-policy self-distillation for reasoning, 2026. URL https://arxiv.org/abs/2605.21606.

Leyi Pan, Shuchang Tao, Yunpeng Zhai, Lingzhe Zhang, Zhaoyang Liu, Bolin Ding, Aiwei Liu, and Lijie Wen. RLCSD: reinforcement learning with contrastive on-policy self-distillation. *CoRR*, abs/2606.11709, 2026. doi: 10.48550/ARXIV.2606.11709. URL https://doi.org/10.48550/arXiv.2606.11709.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

QwenTeam. Qwen3 technical report. *CoRR*, abs/2505.09388, 2025. doi: 10.48550/ARXIV.2505.09388. URL https://doi.org/10.48550/arXiv.2505.09388.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *CoRR*, abs/2402.03300, 2024. doi: 10.48550/ARXIV.2402.03300. URL https://doi.org/10.48550/arXiv.2402.03300.

Hong Wang, Zhezheng Hao, Jian Luo, Chenxing Wei, Yao Shu, Lei Liu, Qiang Lin, Hande Dong, and Jiawei Chen. Scheduling your LLM reinforcement learning with reasoning trees. *CoRR*, abs/2510.24832, 2025a. doi: 10.48550/ARXIV.2510.24832. URL https://doi.org/10.48550/arXiv.2510.24832.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xionghui Chen, Jianxin Yang, Zhenru Zhang, Yuqiong Liu, An Yang, Andrew Zhao, Yang Yue, Shiji Song, Bowen Yu, Gao Huang, and Junyang Lin. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for LLM reasoning. In Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*, 2025b. URL http://papers.nips.cc/paper_files/paper/2025/hash/a797c2d2e0c1fdabf4d1ab8cd0b465c6-Abstract-Conference.html.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=1PL1NIMMrw.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/9d5609613524ecf4f15af0f7b31abca4-Abstract-Conference.html.

Yuanda Xu, Hejian Sang, Zhengze Zhou, Ran He, Zhipeng Wang, and Alborz Geramifard. TIP: token importance in on-policy distillation. *CoRR*, abs/2604.14084, 2026. doi: 10.48550/ARXIV.2604.14084. URL https://doi.org/10.48550/arXiv.2604.14084.

Chenxu Yang, Chuanyu Qin, Qingyi Si, Minghui Chen, Naibin Gu, Dingyu Yao, Zheng Lin, Weiping Wang, Jiaqi Wang, and Nan Duan. Self-distilled RLVR. *CoRR*, abs/2604.03128, 2026. doi: 10.48550/ARXIV.2604.03128. URL https://doi.org/10.48550/arXiv.2604.03128.

Zhicheng Yang, Zhijiang Guo, Yinya Huang, Yongxin Wang, Dongchun Xie, Yiwei Wang, Xiaodan Liang, and Jing Tang. Depth-breadth synergy in RLVR: unlocking LLM reasoning gains with adaptive exploration. *CoRR*, abs/2508.13755, 2025. doi: 10.48550/ARXIV.2508.13755. URL https://doi.org/10.48550/arXiv.2508.13755.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/271db9922b8d1f4dd7aaef84ed5ac703-Abstract-Conference.html.

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 Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*, 2025. URL http://papers.nips.cc/paper_files/paper/2025/hash/a4277440d50f1f15d2cb4c14f7e0c0d2-Abstract-Conference.html.

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-distilled reasoner: On-policy self-distillation for large language models. *CoRR*, abs/2601.18734, 2026a. doi: 10.48550/ARXIV.2601.18734. URL https://doi.org/10.48550/arXiv.2601.18734.

Ziqi Zhao, Xinyu Ma, Liu Yang, Yujie Feng, Daiting Shi, Jingzhou He, Xin Xin, Zhaochun Ren, and Xiao-Ming Wu. Rosd: Reflective on-policy self-distillation for language model reasoning across domains, 2026b. URL https://arxiv.org/abs/2605.28014.

Binbin Zheng, Xing Ma, Yiheng Liang, Jingqing Ruan, Xiaoliang Fu, Kepeng Lin, Benchang Zhu, Ke Zeng, and Xunliang Cai. SCOPE: signal-calibrated on-policy distillation enhancement with dual-path adaptive weighting. *CoRR*, abs/2604.10688, 2026. doi: 10.48550/ARXIV.2604.10688. URL https://doi.org/10.48550/arXiv.2604.10688.

A COMPLETE SPECIFICATION OF LBSD

A.1 END-TO-END TRAINING PROCEDURE

For each training example, LBSD samples a main completion from the student context, while the teacher context additionally includes the extracted solution strategy. Algorithm 1 summarizes the order of main-path scoring, local continuation sampling, branch weighting, and selective distillation.

The supplied launch configurations set $B = 3$, $K = 2$, and $\alpha = 0.5$, and enable token selection on both the main path and branch suffixes. These configurations disable progressive branching and use neither a main-path-only warmup nor a branch-budget ramp.

A.2 REASONING-STEP AND BRANCH-POINT CONSTRUCTION

Step segmentation. The completion text is recursively partitioned with the delimiter priority `\n\n`, `\n`, `.`, `,`, `;`. A substring whose tokenization contains at most $L_{\max} = 60$ tokens is retained without further splitting; otherwise, the procedure tries the next delimiter in the priority order and recursively processes the resulting nonempty pieces, retaining each delimiter with the preceding piece. When all delimiters have been exhausted, the procedure uses tokenizer offsets to cut after the first $L_{\max}$ tokens; if offsets are unavailable or that cut is invalid, it bisects the character string and continues recursively. With an offset-producing tokenizer, the resulting character spans are mapped to token spans in the tokenization of the full completion.

Token utility and step scoring. Rank normalization is performed separately for entropy and discrepancy over the main completion. For a vector of length $T > 1$, values are sorted in ascending order with ties ordered by their original positions, and the normalized rank is the zero-based position in that ordering divided by $T - 1$; a singleton receives rank zero. The normalized signals are combined using Eq. (2), with $\alpha = 0.5$. Writing $Q_{0.8}(U)$ for the linearly interpolated empirical 0.8 quantile of main-path utilities, the score of a step with token span $[a_j, b_j)$ is

$$S_j = \sum_{t=a_j}^{b_j-1} \mathbf{1}\{U_t \geq Q_{0.8}(U)\}.$$

The threshold is inclusive, so tied utilities at the threshold need not yield exactly 20% high-utility positions.

Step selection and branch prefixes. In the supplied configurations, all step positions are eligible and the minimum step length is one token. Eligible steps are ordered by decreasing S_j, with ties resolved in favor of the earlier step, and at most three steps are selected. The prefix for a selected step ends immediately before that step's first token, so each local continuation regenerates the selected step rather than retaining it as context. Branch continuations are sampled from selected prefixes of the original main completion, without recursively expanding the generated branches. The supplied scripts limit the main completion to 1,024 generated tokens; a branch receives a generation budget of $\max(0, 1024 - \text{prefix length})$, and no request is made when this budget is zero.

A.3 NORMALIZATION AND LOSS CONVENTIONS

Active regions and token normalization. The active region is the entire completion for the main path and only the newly generated suffix for a branch path. Entropy and discrepancy are independently min–max normalized within each active region, using

$$\widetilde{z}_t = \begin{cases} 0, & \max_{u \in R} z_u - \min_{u \in R} z_u \leq 10^{-8}, \\ \dfrac{z_t - \min_{u \in R} z_u}{\max_{u \in R} z_u - \min_{u \in R} z_u}, & \text{otherwise}, \end{cases} \qquad z \in \{H, D\},$$

where R denotes that path's active region. The normalized signals are inserted into Eq. (5) to obtain token importance. For a nonempty active region containing N positions, the number retained is

Algorithm 1 Local-branch selective self-distillation

Require: Problem x, extracted strategy s^*, student policy, teacher evaluator, branch-point budget B, continuation budget K

1: Sample a main completion under $C_s(x)$ and segment its text into reasoning steps (Section A.2).
2: Evaluate main-path entropy and discrepancy; rank-normalize both signals and combine them using Eq. (2).
3: Score eligible steps by high-utility token counts and select up to B branch points.
4: **for** each selected step **do**
5: Retain the prefix preceding its first token and sample K student continuations.
6: **end for**
7: Compare main and sampled continuations over a local window to obtain branch alignment scores and weights.
8: **for** each training path **do**
9: Compute importance within the active region and retain its highest-scoring positions (Section A.3).
10: Average the token loss over retained positions, stopping gradients through teacher predictions.
11: **end for**
12: Average path losses within each surviving branch group; combine weighted group losses with the main-path loss and average across examples.

$$n_{\text{keep}} = \min\{N, \max\{8, \lceil 0.8N \rceil\}\},$$

and the retained positions are those returned by the top-n_{keep} importance selection. Consequently, regions shorter than eight tokens are retained in full, and the retained fraction can exceed 80% for short regions. Student and teacher predictions are aligned to the same completion positions using their respective prompt lengths, and shared branch-prefix positions do not contribute directly to the branch loss.

Scoring distributions and local comparison windows. In the supplied configurations, entropy is computed from the full student vocabulary distribution, and the discrepancy used for branch selection, branch alignment, and token selection is full-vocabulary forward KL at distillation temperature 1. For distillation, we compute forward KL at temperature 1 on the teacher's top-20 vocabulary support after renormalization and clip pointwise KL contributions at 0.05. Branch alignment compares the main and branch continuations over at most 128 tokens beginning at the selected step's first token, averaging over the available finite discrepancy values within each window. The implementation aggregates the continuation-window means with a trimmed mean; for the configured two continuations and trim fraction 0.2, no values are trimmed, so this aggregation reduces to the arithmetic mean in Eq. (4).

Objective aggregation. Each path loss is normalized by its number of retained positions, rather than by its unmasked completion length. The supplied scripts set the branch-loss coefficient to $\lambda = 1.2$. The implementation treats branch weights as fixed numerical coefficients when differentiating the distillation objective.

B Privileged Strategy Construction

B.1 Strategy extraction protocol

For each training problem, LBSD constructs an abstract solution strategy from the problem and an available reference reasoning trajectory using Qwen3.5-27B. We retain DeepMath examples with difficulty strictly greater than 7.0. The extractor receives the problem and the reference reasoning trajectory, while the separate ground-truth answer field is not included in the extraction prompt.

The strategy is generated with temperature 0.2, top-p 0.95, and repetition penalty 1.05. If the generated response contains a </think> marker, only the content following the marker is retained as the final strategy. No additional programmatic leakage filter is applied after generation.

B.2 PROMPT TEMPLATES

The prompts below reproduce the templates used for strategy extraction and LBSD training. The strategy-extraction prompt is applied offline. During training, student rollout generation disables the Qwen3 thinking mode, whereas privileged-teacher evaluation enables it. Each prompt is rendered with the corresponding model's chat template.

Strategy extraction prompt. Qwen3.5-27B receives a problem and its reference reasoning trajectory and produces an abstract problem-solving strategy. The prompt explicitly instructs the extractor to omit the final answer, complete derivations, and answer-determining intermediate information.

Listing 1: System prompt for strategy extraction.

```
You are a mathematical strategy distillation assistant. Your task is to
    read a problem and a reference reasoning trajectory, then extract a
    high-level solution strategy. The strategy should guide another model
     toward solving the problem, but it must not reveal the final answer
    or provide a step-by-step solution.
```

Listing 2: User prompt for strategy extraction.

```
You will be given a math problem and a reference reasoning trajectory.

Your task is to write a high-level solution strategy for the problem.

The strategy should help another model understand how to approach the
    problem, but it should not allow the model to solve the problem by
    merely copying your text. The goal is to provide conceptual guidance,
     not a worked solution.

Requirements:
1. Do not give the final answer.
2. Do not include a complete proof or a step-by-step derivation.
3. Do not copy sentences, equations, or detailed procedures from the
    reference reasoning trajectory.
4. Do not reveal key intermediate numerical results, final simplified
    expressions, or decisive case conclusions.
5. Focus on the main idea, useful viewpoint, relevant concepts, and why
    this approach is natural.
6. Describe the type of reasoning that should be used, not the exact
    sequence of operations.
7. Mention important obstacles or traps only at a conceptual level.
8. Keep the strategy concise and abstract.
9. The output should be useful as a hint or planning guide, not as a
    solution.

Output format:
- Problem type:
- Core intuition:
- Strategic direction:
- Key tools or concepts:
- What to avoid:

[Problem]
{problem}

[Reference reasoning trajectory]
{reference_solution}
```

Student prompt. The student receives only the problem statement.

Listing 3: Student prompt used during training.

```
Problem: {problem}

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

Privileged-teacher prompt. The privileged teacher receives the same problem together with the extracted abstract strategy. Neither the original reference reasoning trajectory nor the separate ground-truth answer field is directly included in this training-time teacher prompt.

Listing 4: Privileged-teacher prompt used during training.

```
Problem: {problem}

Here is a solution idea for this problem:
=== Solution Idea Begin ===
{solution_strategy}
=== Solution Idea End ===

After reading the solution idea above, make sure you truly understand the
     strategy, key concepts, and logical flow behind it. Do not copy its
    wording or mechanically follow its structure. Now, use this idea only
     as guidance, and solve the problem above in your own words.

Present a complete step-by-step solution, carry out any necessary
    calculations yourself, justify each important step, and make the
    answer clear and self-contained.

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

C Implementation and Evaluation Details

The common model, dataset, and benchmark settings are described in Section 4.1. This section reports the remaining optimization and evaluation details required to reproduce the experiments.

Baseline implementations. GRPO and OPSD are reproduced using the released OPSD implementation, while RLSD and RLCSD are reproduced using their respective open-source implementations. All baselines are adapted to our model, data, and evaluation interfaces. GRPO is trained for 500 optimization steps, while OPSD is trained for 100 optimization steps. RLSD is trained for 100 optimization steps using four rollouts per problem and an effective batch size of 64. RLCSD is trained for 60 optimization steps. All baselines use LoRA adaptation. GRPO and OPSD follow the training configurations of their released implementations, while RLSD and RLCSD are trained for approximately 10 hours or longer on $8\times$ NVIDIA H20 GPUs. All primary benchmark results are reported from the final checkpoint of each method's training run. The fixed-checkpoint protocol avoids method-dependent checkpoint selection.

LBSD configuration. LBSD is trained for 75 optimization steps using LoRA adaptation. The student uses the LoRA-adapted model, whereas teacher logits are computed with the LoRA adapter disabled and treated as stop-gradient targets. Each main trajectory is expanded at three selected branch points, with two local continuations sampled from each point. Student entropy and teacher–student discrepancy receive equal weight in branch-point scoring, corresponding to $\alpha = 0.5$. Selective distillation retains 80% of the active token positions, and the local-branch loss coefficient is set to $\lambda = 1.2$. The main experiments use forward-KL distillation over the teacher's top-20 logits. Each run uses eight NVIDIA H20 GPUs and takes approximately ten hours.

Evaluation configuration. For each benchmark problem, we independently sample 36 responses from the evaluated checkpoint. We report the mean correctness over all sampled responses, which we denote as Avg@36:

Table 5: Key training configuration for LBSD.

Configuration	Value
Learning rate	5×10^{-6}
Effective batch size	64
Total training steps	75
LoRA rank / scaling	64 / 128
Maximum completion length	1,024 tokens
Branch points B	3
Continuations per branch point K	2
Branch-score mixture α	0.5
Token keep ratio	0.8
Distillation temperature	1.0
Teacher-logit top-k	20
Per-token divergence clip	0.05
Branch-loss coefficient λ	1.2
Primary comparison checkpoint	Step 75
Compute	$8\times$ NVIDIA H20, $\sim$10 hours

Table 6: Evaluation configuration.

Configuration	Value
Samples per problem	36
Thinking mode	Enabled
Temperature	1.0
Top-p	0.95
Maximum new tokens	38,912
Required answer format	Final answer in `\boxed{}`
Answer extraction	Last valid `\boxed{}` expression
Equivalence checker	`math-verify`

$$\text{Avg@36} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{36}\sum_{j=1}^{36}\mathbb{I}[\hat{y}_{i,j} \equiv y_i]\,, \tag{8}$$

where N is the number of benchmark problems, $\hat{y}_{i,j}$ denotes the answer extracted from the j-th sampled response to problem i, and y_i is the reference answer. Avg@36 measures the average correctness of independently sampled responses and is distinct from both pass@k and majority-vote accuracy.

The evaluator extracts the final valid boxed expression from each response and checks symbolic equivalence against the reference answer using `math-verify`. If symbolic parsing fails, the evaluator falls back to normalized string comparison. Responses without a valid boxed expression are counted as incorrect.

Leakage and response-style metrics. For the privileged-context analysis in Section 4.4, we sample 1,200 rollouts with a maximum generation length of 1,024 tokens, matching the training-time rollout limit. Leakage Rate is computed by manually inspecting the sampled responses and reporting the percentage that reveal privileged information unavailable to the student. To quantify broader changes in generation behavior, we compute four response-style statistics on the same rollouts. Answer Density and Solution Density measure the occurrences of the case-insensitive whole words `answer` and `solution`, respectively, normalized to occurrences per 1,000 response words and then averaged across responses. Boxed Freq. and Final-Answer Freq. measure the average number of `\boxed{...}` markers and occurrences of the case-insensitive phrase `final answer` per response, respectively. Table 7 reports the raw statistics before converting the response-style metrics to deviations from the Base model. In Table 2, we report the absolute deviation of each response-

Table 7: Raw privileged-context leakage and response-style statistics of Qwen3-4B on DeepMath and AIME26.

	Leakage	Response Style			
Method	Rate (%)	Answer Density	Solution Density	Boxed Freq.	Final-Answer Freq.
DeepMath					
Base	1.00	0.6179	0.1243	0.1325	0.0392
OPSD	1.25	1.6014	0.2965	0.6675	0.1333
LBSD	0.75	1.1833	0.1561	0.5508	0.1925
AIME26					
Base	0.92	0.2782	0.2609	0.0417	0.0258
OPSD	1.25	0.5959	0.4384	0.1775	0.0808
LBSD	0.67	0.2900	0.2895	0.0342	0.0308

style statistic from the corresponding Base model so that smaller values indicate behavior closer to the original model.