KnowRL: Boosting LLM Reasoning via Reinforcement Learning with Minimal-Sufficient Knowledge Guidance

Anonymous ACL submission

Abstract

Reinforcement learning with verifiable rewards (RLVR) improves large language model reasoning, but remains limited by sparse rewards on hard problems. We propose **KnowRL**, a knowledge-guided RL framework that alleviates reward sparsity by injecting minimal-sufficient knowledge points (KPs) instead of long solution prefixes or full templates. KnowRL decomposes correct solutions into atomic KPs, filters leakage and redundancy, and selects compact problem-specific subsets through interaction-aware offline evaluation. We further introduce Constrained Subset Search (CSS) to account for pruning interaction paradox and identify robust KP subsets for RL training. Across eight mathematical reasoning benchmarks, KnowRL improves both 1.5B and 7B backbones. Without test-time KP hints, KnowRL-Nemotron-1.5B and KnowRL-DS-Distill-Qwen-7B reach 70.08 and 65.94 average accuracy, respectively, establishing new state-of-the-art results; with CSS-selected KPs, they further improve to 74.15 and 68.87. Out-of-domain evaluations further indicate that KnowRL improves the underlying reasoning policy rather than merely exploiting math-specific hints. Multiple ablation studies further validate the effectiveness of KnowRL. The model, curated data, and code will be released.

1 Introduction

RLVR has emerged as a paradigm for improving LLM reasoning by optimizing verifiable correctness (Team, 2026; Guo et al., 2025a; Wang et al., 2026; Team, 2025). By aligning outputs with rule-based verifiers, RLVR provides scalable supervision without relying on human preference annotations. However, RLVR suffers from a key bottleneck: *reward sparsity* on difficult samples. For complex reasoning tasks, LLMs often produce uniformly incorrect rollouts, yielding zero advantage

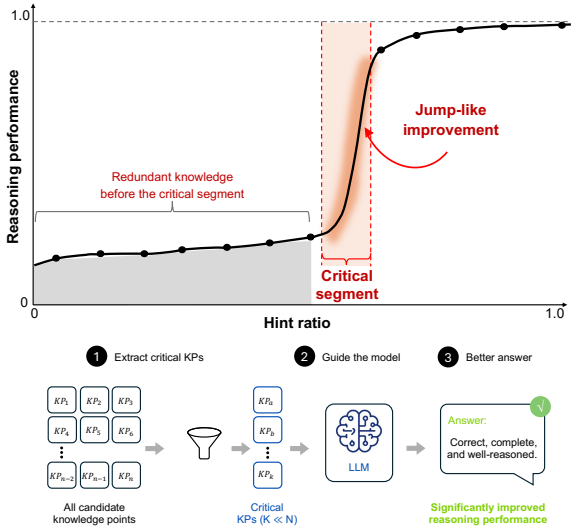


Figure 1: Overview of the critical-segment effect and the resulting compact knowledge-point selection workflow.

under group-based optimization methods such as GRPO (Shao et al., 2024). Consequently, a large portion of training data fails to contribute gradients, reducing learning efficiency.

To address this issue, recent work introduces *hint-based RL*, which injects auxiliary guidance into prompts to increase the probability of generating reward-yielding responses. Existing approaches can be categorized into three types: (1) **fixed-ratio solution-prefix hints** that prepend a partial reference solution using a predefined, constant hint ratio across training (e.g., QuestA (Li et al., 2025), POPE (Qu et al., 2026)); (2) **adaptive solution-based hints** that dynamically determine the hint ratio based on instance difficulty or training state (e.g., StepHint (Zhang et al., 2025b), UFT (Liu et al., 2025a)); and (3) **abstraction-based hints** that provide reasoning templates or conceptual abstractions generated by teacher models (e.g., TAPO (Wu et al., 2025), Guide (Nath et al., 2025), Scaf-GRPO (Zhang et al., 2025c)).

Despite their differences, existing hint-based RL

methods implicitly assume that stronger guidance requires longer solution prefixes or richer abstractions. However, our analysis reveals three observations that motivate compact knowledge guidance. First, we observe a *critical-segment effect*: as shown in Figure 1, performance does not increase proportionally with hint ratio. Instead, accuracy exhibits a sharp jump once a key segment appears, followed by diminishing gains. This indicates that only a small set of knowledge components is sufficient to shift the policy toward reward-yielding trajectories, motivating us to locate the critical interval and curate compact KP subsets from complete solutions. Appendix A further visualizes this jump across varying hint ratios on 50 randomly sampled training instances. Second, we identify *cross-hint inconsistency*: longer prefixes or abstract templates may introduce branching and conceptual ambiguity, complicating policy updates. Third, we observe a trade-off between guidance independence and training efficiency. Abstraction-based hints often rely on teacher-generated guidance, interrupting online RL and increasing computational cost.

Together, these findings suggest that the core challenge is not providing *more* guidance, but selecting *minimal, coherent knowledge units* that are sufficient to overcome reward sparsity. This naturally raises a fundamental question: *can models be effectively trained using* minimal yet sufficient hints *that unlock rewards without introducing redundant guidance?*

To this end, we propose **Knowledge-Guided Reinforcement Learning (KnowRL)**, a new hint injection approach for hint-based RL that formulates hint design as a *minimal sufficient guidance problem*. Rather than injecting long solution prefixes or full reasoning templates, KnowRL decomposes guidance into atomic knowledge points (KPs) and identifies the minimal subset required to unlock reward learning. Importantly, we model a *pruning interaction paradox*: removing a single KP may improve accuracy, while removing multiple such "bad" KPs together can reduce accuracy due to inter-KP dependencies. This paradox further guides our KP selection strategy design.

We construct KP hints for the training set through a structured pipeline and explore multiple KP selection strategies. Our final method adopts Constrained Subset Search (CSS), which prunes first and then performs global search over the remaining candidates, achieving the best performance with the fewest KPs. In practice, simple problems receive no hints, while minimal KP subsets are injected only for harder samples during training. The resulting RL-trained model achieves new state-of-the-art results across eight benchmarks with or without KP hints at inference, indicating that effective guidance depends on critical knowledge structure rather than long prefixes or heavy abstraction templates.

In summary, our contributions are threefold. **First**, we introduce KnowRL, a new hint injection approach for hint-based RL from a minimal-sufficiency perspective, and empirically demonstrate a non-linear, jump-like performance pattern (critical-segment effect), revealing that effective guidance depends on selectively injecting key knowledge rather than cumulatively increasing hint length. **Second**, we design several KP selection pipelines that ensure minimal, non-redundant, and interaction-compatible KP subsets for each problem, and through detailed comparative analyses identify CSS as the optimal selection strategy. **Third**, we conduct experiments across models of different scales and achieve state-of-the-art scores across benchmarks, with comprehensive ablation studies analyzing how different data selection strategies and KP extraction models affect the final RL training outcomes.

2 Related Work

Solution-Prefix Hints These methods typically extract fixed proportions of solution prefixes to guide the model. QuestA (Li et al., 2025) and Hint (Wang et al., 2025) augment hard prompts with a fixed p% solution prefix, while POPE (Qu et al., 2026) further refines this approach by optimizing prefix selection based on token-level importance scores, but retains the core characteristic of fixed-ratio truncation.

Adaptive Solution-Based Hints To overcome the rigidity of fixed-ratio prefixes, subsequent works introduce adaptivity into solution-based hints, refining how and when guidance is injected. GHPO (Liu et al., 2025c), G²RPO-A (Guo et al., 2025b) and Hint-GRPO (Huang et al., 2025a) scale hint length with task difficulty or recent reward signals, while StepHint (Zhang et al., 2025b) refines granularity by partitioning reasoning chains into semantic steps for multi-level control. ADHint (Zhang et al., 2025a) further incorporates offline difficulty priors to pre-calibrate hint strength.

Alongside these adaptive refinements, a paral-

lel line of work incorporates solution prefixes into hybrid SFT–RL pipelines. BREAD (Zhang et al., 2025d) ensures at least one successful trajectory per update by increasing the proportion of expert prefixes upon failure; Prefix-RFT (Huang et al., 2025b) concatenates offline SFT prefixes with online RL continuations to produce hybrid rollouts; and UFT (Liu et al., 2025a) employs a cosine-annealing schedule to progressively reduce hint length during training.

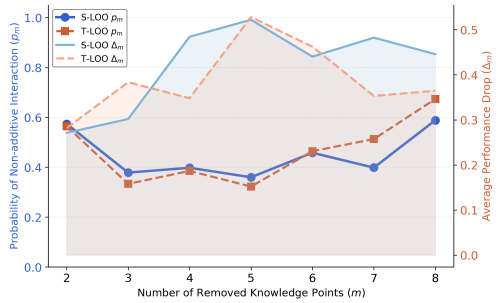
Abstraction-Based Hints Abstraction-based hints shift guidance from solution prefixes to high-level concepts, principles, and structured reasoning patterns rather than partial solutions. Guide (Nath et al., 2025) introduces natural-language hints generated by stronger teacher models (e.g., GPT-4o) to accelerate learning on hard problems; Scaf-GRPO (Zhang et al., 2025c) proposes a two-stage scaffold injection, where abstractions are generated by DeepSeek. Complementing this line, TAPO (Wu et al., 2025) incorporates structured "thought patterns" as external templates that encode general reasoning strategies.

More recently, abstraction generation itself has been incorporated into the learning objective. Self-Hinting (Liao et al., 2026) enables the model to act as its own teacher: given a solution, it generates abstract hints to guide subsequent rollouts, reducing reliance on external teachers. RLAD (Qu et al., 2025) further refines this idea by training models with auxiliary supervision to produce higher-quality abstractions during RL.

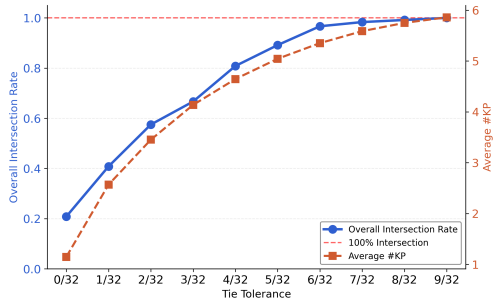
These typically depend on strong teacher models or carefully designed templates, and overly vague abstractions may fail to provide actionable signals for difficult reasoning tasks.

3 KnowRL

In this section, we present KnowRL from a framework perspective. At a high level, KnowRL follows a simple end-to-end workflow: for each training problem, it first constructs candidate knowledge points (KPs), then removes leakage and redundancy to obtain a compact problem-specific subset, and finally uses the curated subset as hint data for RL training only when guidance is needed. Appendix B.1 provides an example of how these KPs are used during training.



(a) Pruning interaction paradox under LOO-style selection strategies.



(b) Tolerance-threshold sensitivity.

Figure 2: Interaction-aware KP selection: inconsistency-induced degradation and the δ–compactness trade-off.

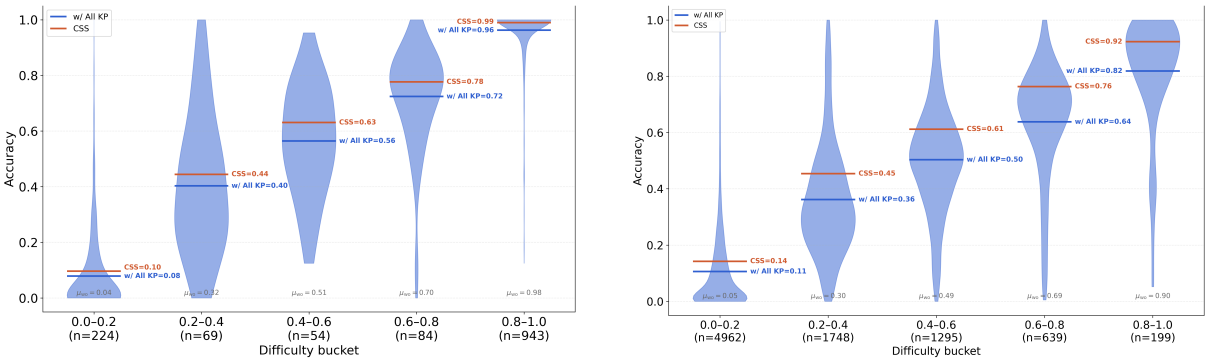
3.1 KP Curation

The first stage of KnowRL constructs candidate KP annotations from datasets that already contain verified correct solutions. We extract KPs from these correct reasoning traces and then verify each candidate to ensure that retained KPs are generalizable and do not leak the final answer or instance-specific information. Details of the curation pipeline and prompts are provided in Appendix B.2.

3.2 Problem-wise KP Subset Selection

For a problem with candidate KP set $\mathcal{K}$, we estimate offline accuracies under different configurations: $A_\emptyset$, $A_\mathcal{K}$, and $A_{-i} = A(\mathcal{K} \setminus \{k_i\})$. Here, A_{-i} corresponds to leave-one-out ablation of k_i, allowing us to quantify the marginal importance of each KP by measuring the performance drop when it is removed.

A straightforward and practical strategy is "Max-Score", selecting the configuration achieving the highest accuracy among $\{\emptyset, \mathcal{K}, \mathcal{K} \setminus \{k_i\}\}$. While effective (Table 1), this strategy restricts each problem to choosing from only three types of configurations: using no KPs, using the full set $\mathcal{K}$, or removing exactly one KP (i.e., an $n-1$ subset). This coarse search space can cause mismatches: problems that benefit from fewer KPs may be as-



(a) Test-set comparison across difficulty levels.

(b) Training-set comparison across difficulty levels.

Figure 3: Difficulty-bucket analysis on both test and training sets, where buckets are defined by no-KP accuracy. Full-KP injection shifts the violin distributions upward and improves mean performance in most buckets, but it also induces regressions on a subset of instances. In contrast, CSS-selected KPs deliver larger and more consistent gains across buckets. On the x-axis, $(n = \cdot)$ denotes the number of samples in each bucket, and the gray marker μ_{wo} indicates the no-KP mean accuracy of that bucket.

signed the full set. Consequently, the resulting selections can be suboptimal.

3.2.1 S-LOO and T-LOO

We unify KP selection as a parameterized decision operator whose goal is to choose the most beneficial KP configuration for each problem, reducing dependence on KPs while preserving performance. To this end, we introduce a tolerance parameter $\varepsilon \geq 0$, which controls how strictly we treat borderline cases when selecting the optimal configuration.

Given ε, the generalized selection strategy is formalized as a mapping $\Phi_\varepsilon : \mathcal{K} \longrightarrow \mathcal{K}^* \subseteq \mathcal{K}$, where $\mathcal{K}^*$ denotes the final selected subset and, letting $A_{\max} = \max_i A_{-i}$ and $M_\varepsilon = A_{\max} - \varepsilon$, is defined by

$$\Phi_\varepsilon(\mathcal{K}) = \begin{cases} \emptyset, & \text{if } A_\emptyset \geq \max(A_\mathcal{K},\, M_\varepsilon), \\ \mathcal{K}, & \text{if } A_\mathcal{K} > \max(A_\emptyset,\, M_\varepsilon), \\ \mathcal{K} \setminus S, & \text{otherwise,} \end{cases}$$

where $S = \{k_i \mid A_{-i} \geq \max(A_\mathcal{K}, A_\emptyset) - \varepsilon\}$.

Within this framework, different strategies correspond to different choices of ε. When $\varepsilon = 0$, we obtain Strict Leave-One-Out selection (S-LOO). Since accuracy estimates are based on finite sampling and thus subject to randomness, we further introduce a tolerance band $\varepsilon = 1/32$, yielding Tolerant Leave-One-Out selection (T-LOO). Compared to S-LOO, T-LOO allows up to one-sample-scale performance rollback in near-tie cases, making selection more stable on borderline problems.

As shown in Table 1, S-LOO and T-LOO select substantially fewer KPs than Max-Score, but they also yield lower accuracy. A major reason is that LOO-based pruning overgeneralizes from single-KP ablations: even when removing k_i alone improves accuracy, jointly removing all KPs marked as "non-essential" can cause a performance drop rather than an improvement. We refer to this phenomenon as the *pruning interaction paradox*.

To quantify this effect, we characterize cases where removing each of m KPs individually improves performance, but removing them jointly degrades performance. We define the positive-contribution set: $\mathcal{K}^+ = \{k_i \mid A_{-i} \geq \max(A_\mathcal{K}, A_\emptyset)\}$. For subsets $S \subseteq \mathcal{K}^+$ with $|S| = m$, define: $A_{\text{joint}}(S) = A(\mathcal{K} \setminus S)$ and $\bar{A}_{\text{single}}(S) = \frac{1}{m}\sum_{k_i \in S} A_{-i}$. Across problems, we compute: $p_m = \Pr(A_{\text{joint}}(S) < \bar{A}_{\text{single}}(S))$ and $\Delta_m = \mathbb{E}[\bar{A}_{\text{single}}(S) - A_{\text{joint}}(S) | A_{\text{joint}}(S) < \bar{A}_{\text{single}}(S)]$. As summarized in Figure 2a, cross-hint inconsistency occurs frequently (typically $p_m \in [40\%, 60\%]$), with substantial performance drops.

3.2.2 Constrained Subset Search (CSS)

To address this pruning interaction paradox, a theoretically optimal approach would evaluate all 2^n KP subsets, but this is computationally infeasible.

We instead construct a constrained search space. Define $\mathcal{H} = \{k_i \mid A_{-i} \geq \max(A_\mathcal{K}, A_\emptyset)\}$ as the non-degrading KPs and $\mathcal{N} = \{k_i \in \mathcal{H} \mid A_{-i} \geq A_{\max}\}$ as near-optimal removals. KPs in $\mathcal{N}$ can be removed directly, since deleting them yields substantially improved performance. Moreover, since $|\mathcal{N}|$ is small on average (1.21), removing $\mathcal{N}$ alone rarely triggers the pruning interaction paradox.

Let $\mathcal{C} = \mathcal{H} \setminus \mathcal{N}$. We enumerate subsets only

Selection Strategy	AIME24	AIME25	BRUMO25	HMMT Feb 25	AMC23	CMIMC25	MATH-500	Olympiad Bench	Avg.	Avg. #KP
w/o KP	58.75	48.44	61.67	30.10	90.55	30.08	92.40	71.70	60.46	0.00
All KP	60.94	49.06	61.15	32.50	89.69	32.34	92.23	70.55	61.06	5.86
Random	60.52	49.27	61.04	33.23	91.02	31.09	91.65	71.88	61.21	2.53
Max-Score	62.60	49.79	64.27	34.79	90.94	32.97	92.53	73.89	62.72	2.61
S-LOO	62.71	49.27	63.85	33.54	91.72	33.52	92.90	73.70	62.65	1.72
T-LOO	62.08	49.27	64.17	33.65	91.25	33.67	92.40	73.46	62.49	1.20
CBRS	63.02	49.90	64.17	34.79	91.56	33.59	92.65	73.89	62.95	2.60
CSS	**64.48** ↑5.73	**50.52** ↑2.08	**65.00** ↑3.33	**35.73** ↑5.63	**91.72** ↑1.17	**36.72** ↑6.64	**92.90** ↑0.50	**74.11** ↑2.41	**63.90** ↑3.44	2.57

Table 1: Offline KP selection strategies on Nemotron-1.5B, used to compare which data selection strategy is more effective under offline evaluation and to approximate its potential effectiveness for RL training. Avg.#KP denotes the average number of selected key knowledge points per problem. Up arrows indicate improvements over w/o KP.

within $\mathcal{C}$, yielding search space size $2^{|\mathcal{C}|}$, which is tractable in practice. The final configuration is chosen via: $S^* = \arg\max_S A(S)$, over all constrained candidates plus $\emptyset$ and $\mathcal{K}$. As displayed in Table 1, CSS achieves the best overall tradeoff: higher accuracy (63.90 eight-task average) with only 2.57 KPs per problem.

3.2.3 Consensus-Based Robust Selection (CBRS)

In CBRS, we sample each configuration multiple times to select configurations that exhibit more stable performance across runs.

For run j, define near-optimal configurations:

$$\mathcal{O}^{(j)} = \{c \mid A^{(j)}(c) \geq \max_{c'} A^{(j)}(c') - \delta\},$$

with $\delta = 1/32$. We define robust consensus:

$$\mathcal{O}^* = \begin{cases} \bigcap_{j=1}^{8} \mathcal{O}^{(j)}, & \text{if non-empty,} \\ \arg\max_c \sum_j \mathbf{1}(c \in \mathcal{O}^{(j)}), & \text{otherwise.} \end{cases}$$

Further, when the above rules still yield multiple tied candidates, we select the one with the smaller score variance across the eight independent evaluation runs. Specifically, for any candidate configuration $c \in \mathcal{O}^\star$, let its performance variance over the eight evaluation runs be $\mathrm{Var}(c) = \frac{1}{8}\sum_{j=1}^{8}\left(A^{(j)}(c) - \frac{1}{8}\sum_{j=1}^{8} A^{(j)}(c)\right)^2$. We present the effect of selecting different δ values in Appendix C. As shown in Table 1, CBRS also yields strong performance while maintaining compact KP sets.

Summary. We validate the effectiveness of each data selection strategy through offline evaluation before RL training, with the results summarized in Table 1. The table shows that hint effectiveness depends on robust selection rather than hint quantity alone: full-KP input uses many KPs and may hurt some subsets, while random KPs with CSS-like cardinality still underperform CSS and CBRS. Max-Score and LOO remain limited by cross-hint inconsistency and the pruning interaction paradox. CBRS improves robustness through repeated sampling, but CSS performs best by pruning candidates before global search; the two select similarly sized KP sets (around 2.5 KPs on average) with Jaccard similarity 0.70, indicating strategy-specific selections. Figures 3a and 3b further show that CSS-selected KPs yield larger gains than full-KP input across difficulty levels, motivating our final KP selection pipeline.

4 Experiments

In this section, we examine KnowRL from five aspects: training data construction, training setup, evaluation protocol, final performance, and extensive ablation studies.

4.1 Training Data

We used the open-source QuestA dataset (Li et al., 2025). We retained 8.8k training instances after deduplication. Following Section 3, we obtained KPs using the CSS strategy since it yielded more compact KP sets and the best offline performance. As reported in Figure 4, different data selection strategies substantially improve performance while reducing the number of injected KPs on OpenMath-Nemotron-1.5B (Moshkov et al., 2025). We provide the corresponding offline results for DeepSeek-R1-Distill-Qwen-7B in Appendix D.

4.2 Training Setup

We conduct training experiments based on two models: OpenMath-Nemotron-1.5B and DeepSeek-R1-Distill-Qwen-7B.

We set train_batch_size= 256, performed four updates per step, and used a constant learning rate

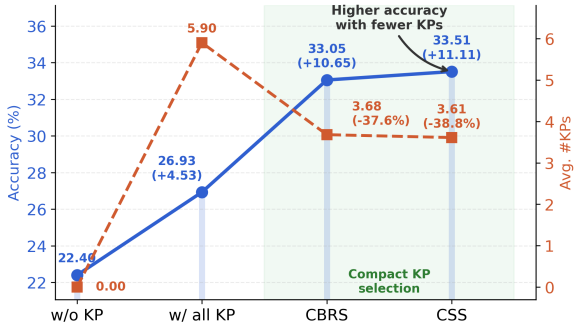


Figure 4: Offline evaluation on the QuestA dataset with different KP selection strategies for OpenMath-Nemotron-1.5B, showing the trade-off between accuracy and the average number of selected KPs.

of 10^{-6} with clip_ratio_range $\in [0.8, 1.28]$. Each question was sampled eight times with top_p= 1.0 and $T = 1.0$, and max_response_length was set to 24k. We used token-mean loss, did not use KL loss or an entropy bonus, and enabled dynamic sampling (Yu et al., 2025). With training config optimizations, KnowRL can be trained economically and effectively on this 4×8 H100 setup. We additionally use entropy annealing to accelerate convergence, with details and ablations provided in Appendix E.

4.3 Evaluation Setup

During training, we used a purely rule-based reward. We evaluate and analyze KP annotations on eight mathematical reasoning benchmarks, detailed information about these benchmarks, including their sources, sizes, and answer formats, is provided in Appendix F. For offline evaluation, we followed the JustRL-style protocol: first applied the rule-based evaluator based on `mathverify==0.8.0`, and when it failed, further verified with CompassVerifier-3B (Liu et al., 2025b). We used a maximum length of 32k tokens, top_p= 0.7, and $T = 0.9$, with 8 samples per problem on MATH-500 and Olympiad-Bench (reported as mean@8) and 32 samples per problem on the remaining benchmarks (reported as mean@32).

4.4 Experiment Results

Overall performance. Across all eight benchmarks, KnowRL consistently achieves the strongest overall performance at both model scales. At the 1.5B scale, KnowRL-Nemotron-1.5B reaches an average score of 70.08 even without KP hints, clearly surpassing Nemotron-1.5B by +9.63 points and outperforming JustRL by +1.50. When incorporating selected KPs, performance further improves to 73.47 with CBRS and 74.16 with CSS, establishing a new state of the art at the 1.5B scale. At the 7B scale, KnowRL-DS-Distill-Qwen-7B shows the same trend: without KP hints, it improves over R1-Distill-Qwen-7B by +9.37 points and reaches 65.94 average accuracy; with selected KPs, it further improves to 68.40 with CBRS and 68.87 with CSS, outperforming the strongest 7B baselines in our comparison. Notably, the substantial no-KP improvements at both scales show that KnowRL improves the underlying policy itself, rather than relying only on test-time hint injection.

The gains are especially clear on competition-style benchmarks. Under CSS selection, KnowRL improves the average score over the corresponding backbone by +10.52 points at 1.5B and +9.27 points at 7B, suggesting that compact KP selection strengthens long-horizon reasoning rather than merely adding superficial hints.

Out-of-domain generalization. To examine whether KnowRL merely fits math-specific hint patterns, we further evaluate KnowRL-Nemotron-1.5B on out-of-domain reasoning benchmarks covering code reasoning, general reasoning, and scientific reasoning. As shown in Table 3, although the backbone OpenMath-Nemotron-1.5B is already math-specialized, KnowRL improves over it on all evaluated out-of-domain tasks. In contrast, QuestA and JustRL show more mixed behavior on GPQA-style benchmarks. These results suggest that KP-guided RL improves the underlying reasoning policy rather than simply exploiting math-specific hint patterns. The corresponding 7B out-of-domain results are provided in Appendix G.

Additional analyses. We provide additional analyses in Appendix H, including the detailed training-budget and offline KP construction cost comparison in Appendix H.1, as well as the training-set correct-count analysis in Appendix H.2.

4.5 Ablation Studies

Selection strategy. Table 1 identifies CSS as the best KP selection strategy through offline evaluation. We further validate this choice with actual RL training by comparing CSS and CBRS under matched training budgets while keeping the number of selected KPs comparable, so that the comparison focuses on selection quality rather than hint quantity. As shown in Figure 5, CSS achieves higher training accuracy and maintains a smoother

Model	Hint Setting	AIME24	AIME25	BRUMO25	HMMT25	AMC23	CMIMC25	MATH	OlyBench	Avg.
1.5B Models										
Nemotron-1.5B (Moshkov et al., 2025)	w/o KP	59.06	48.33	60.70	30.63	90.70	30.10	92.35	71.70	60.45
	CBRS	63.02	48.96	64.14	34.79	91.56	33.54	92.65	73.89	62.82
	CSS	64.06	50.10	65.00	35.73	90.47	36.67	92.90	74.09	63.63
QuestA (Li et al., 2025)	w/o KP	**71.56**	62.08	67.50	40.94	93.44	41.46	92.95	72.27	67.78
	CBRS	74.27	61.98	73.20	43.75	95.08	46.15	93.95	78.45	70.85
	CSS	74.27	65.00	73.75	44.38	95.08	47.60	94.05	78.52	71.58
JustRL (He et al., 2025a)	w/o KP	69.69	62.92	66.88	40.63	**96.02**	41.77	94.15	76.59	68.58
	CBRS	69.79	62.40	70.47	41.77	95.70	44.48	94.85	78.41	69.73
	CSS	70.42	61.46	70.70	41.56	95.55	45.21	94.60	78.67	69.77
KnowRL-Nemotron-1.5B	w/o KP	69.79 ↑10.73	**64.69** ↑16.36	**69.45** ↑8.75	**41.04** ↑10.41	95.55 ↑4.85	**44.17** ↑14.07	**95.70** ↑3.35	**80.23** ↑8.53	**70.08** ↑9.63
	CBRS	**75.52** ↑12.50	**65.00** ↑16.04	**78.36** ↑14.22	**45.00** ↑10.21	**95.78** ↑4.22	**49.27** ↑15.73	**96.45** ↑3.80	**82.34** ↑8.45	**73.47** ↑10.65
	CSS	**74.58** ↑10.52	**65.21** ↑15.11	**78.13** ↑13.13	**48.75** ↑13.02	**95.70** ↑5.23	**52.19** ↑15.52	**96.20** ↑3.30	**82.44** ↑8.35	**74.15** ↑10.52
7B Models										
R1-Distill-Qwen-7B (DeepSeek-AI, 2025)	w/o KP	54.17	38.44	55.94	23.44	90.08	26.72	94.30	69.49	56.57
	CBRS	56.67	41.46	58.75	26.25	90.55	31.17	94.45	71.12	58.80
	CSS	58.44	41.35	59.38	27.08	91.25	33.05	94.65	71.57	59.60
Light-R1-7B-DS (Wen et al., 2025)	w/o KP	59.06	44.27	56.98	27.19	91.64	30.55	94.28	70.57	59.32
	CBRS	58.33	44.79	61.46	32.40	92.34	34.06	95.28	71.87	61.32
	CSS	59.48	46.25	62.92	29.79	92.58	33.83	94.78	72.29	61.49
AReaL-boba-RL-7B (Fu et al., 2025)	w/o KP	61.88	48.33	61.77	29.48	93.28	30.00	95.20	74.54	61.81
	CBRS	62.29	48.44	66.15	32.81	93.67	35.08	95.83	75.07	63.67
	CSS	63.44	49.06	66.04	34.90	94.06	35.00	95.80	75.41	64.21
SkyWork-OR1-7B (He et al., 2025b)	w/o KP	70.21	54.58	62.08	32.19	93.75	33.91	95.63	75.28	64.70
	CBRS	71.25	55.73	66.67	35.63	93.98	39.14	96.20	76.47	66.88
	CSS	71.88	52.60	67.81	35.83	93.28	40.23	95.80	76.82	66.78
AceReason-Nemotron-7B (Chen et al., 2025)	w/o KP	68.96	53.65	66.04	31.98	95.16	37.81	95.53	75.72	65.61
	CBRS	68.65	55.21	70.94	37.29	95.86	**42.73**	96.35	**78.13**	68.15
	CSS	67.40	55.21	**71.46**	**38.75**	95.39	42.19	96.05	77.65	68.01
KnowRL-DS-Distill-Qwen-7B	w/o KP	71.15 ↑16.98	55.31 ↑16.87	66.46 ↑10.52	32.81 ↑9.37	95.94 ↑5.86	34.53 ↑7.81	95.75 ↑1.45	75.59 ↑6.10	65.94 ↑9.37
	CBRS	74.17 ↑17.50	**56.46** ↑15.00	70.63 ↑11.88	38.23 ↑11.98	95.39 ↑4.84	40.00 ↑8.83	96.40 ↑1.95	75.96 ↑4.84	68.40 ↑9.60
	CSS	**74.48** ↑16.04	**56.46** ↑15.11	71.15 ↑11.77	**38.75** ↑11.67	**96.02** ↑4.77	40.63 ↑7.58	**96.50** ↑1.85	77.02 ↑5.45	**68.87** ↑9.27

Table 2: Evaluation results of RL training with CSS-selected KP data under different test-time prompting strategies (with and without KPs). All scores are evaluated using the protocol described in Section 4.3. For QuestA and JustRL, the w/ KP scores are taken directly from the JustRL paper.

clip-ratio trajectory, whereas CBRS shows more aggressive updates in the middle and late stages of training. Detailed checkpoint-level results are provided in Appendix H.6, where CSS also generalizes better at both intermediate checkpoints. These results confirm that the offline-selected CSS strategy provides more reliable KP subsets and more effective training signals than CBRS.

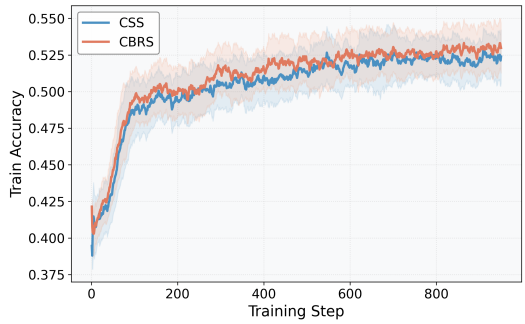
We further compare CSS-selected KP training with full-KP training on the same 1.5B backbone. Full-KP training reaches similar training accuracy but shows weaker no-KP test performance, suggesting overfitting to noisy raw hints. Appendix H.4 provides the detailed analysis and an output example illustrating the resulting fictional-KP phenomenon.

SFT vs. RL. To test whether KnowRL's gains can be explained by simply distilling DeepSeek-R1-generated solutions, we compare KnowRL with two SFT baselines trained on the same 1.5B backbone: one using R1 solutions with KPs and one using R1 solutions without KPs. As shown in Figure 6, both SFT variants remain close to the initial model and substantially below KnowRL-Nemotron-1.5B, while adding KPs to SFT brings no clear benefit over SFT without KPs. This suggests that the improvement does not come merely from exposing the student to teacher-generated solution content; instead, compact selected KPs are most effective when used to guide RL exploration and reinforce reward-yielding trajectories. Detailed SFT settings are provided in Appendix H.5.

KP extractor sensitivity. We further investigate whether KnowRL critically depends on a very strong teacher model for KP extraction. To this end, we replace DeepSeek-R1 with Qwen3-8B during KP extraction, while keeping the subsequent leakage filtering, CSS selection, and offline evaluation protocol unchanged. We focus on the offline utility of the extracted KPs, which is a reasonable proxy in our setting, since our previous analysis shows that offline KP selection quality is strongly

Model	LiveCodeBench pass@1	LiveCodeBench pass@5	MMLU	ARC-Challenge	GPQA-Extended	GPQA-Main
Nemotron-1.5B	0.1137	0.4739	31.79	35.67	29.30	27.46
QuestA	0.1140	0.4740	35.54	39.07	29.85	26.12 ↓1.34
JustRL	0.2464	0.8531	34.27	38.65	27.29 ↓2.01	21.88 ↓5.58
KnowRL-Nemotron-1.5B	**0.5687**	**1.6114**	**37.20**	**40.35**	**29.85**	**27.90**

Table 3: Out-of-domain generalization of KnowRL-Nemotron-1.5B across code, general, and scientific reasoning benchmarks. Down arrows indicate performance drops relative to Nemotron-1.5B.



(a) Training accuracy.

(b) Clip ratio.

Figure 5: Comparison of KP selection strategies under the same training budget.

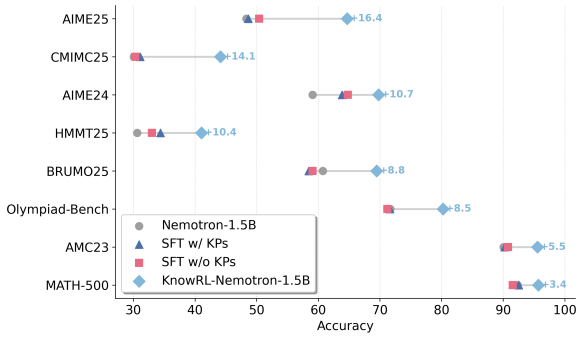


Figure 6: Comparison between SFT baselines trained on DeepSeek-R1-generated correct solutions and RL-based methods.

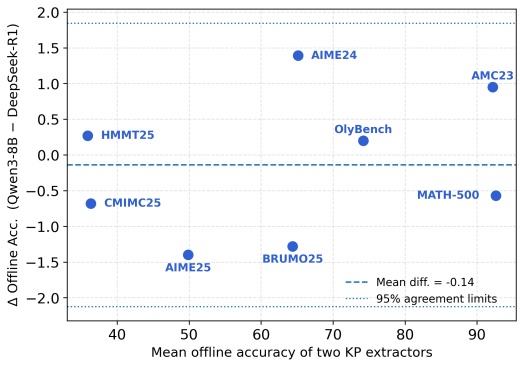


Figure 7: Comparison of offline CSS accuracy using KPs extracted by Qwen3-8B and DeepSeek-R1.

correlated with downstream RL effectiveness.

Figure 7 compares the paired offline CSS scores obtained from DeepSeek-R1 and Qwen3-8B extractors. The mean difference is small (-0.14 points for Qwen3-8B minus DeepSeek-R1), and all benchmarks fall within the 95% agreement limits. The complete benchmark-level results are reported in Appendix H.3. Overall, these results suggest that KnowRL is not tightly coupled to a single strong teacher model and may remain effective with lower-cost KP extractors.

5 Conclusion

We have presented KnowRL, a minimal-sufficient guidance framework for RLVR that converts solution guidance into compact, reusable knowledge-point subsets. KnowRL is motivated by the critical-segment effect and by inter-KP dependencies, and its CSS strategy is first selected through offline evaluation and then validated through actual RL training. Across eight mathematical reasoning benchmarks, KnowRL improves both 1.5B and 7B backbones, achieves strong performance with and without test-time KP hints, and shows improved out-of-domain generalization. These results suggest that compact, structured guidance can serve as an effective scaling principle for sparse-reward RL, motivating future extensions to broader reasoning domains and lower-cost KP curation pipelines.

Limitations

Due to resource constraints, our experiments focus on two representative model scales, 1.5B and 7B. Nevertheless, KnowRL already achieves strong results on these smaller-scale models, demonstrating the effectiveness of our method.

In addition, this work primarily studies mathematical reasoning benchmarks. We are actively exploring the application of KnowRL to datasets from broader domains. In principle, KnowRL can be applied to tasks where problem solving can be improved through explicit knowledge guidance, such as code reasoning and other structured reasoning problems. Extending the framework beyond mathematics is an important direction for future work.

References

Mislav Balunović, Jasper Dekoninck, Ivo Petrov, Nikola Jovanović, and Martin Vechev. 2025. Matharena: Evaluating llms on uncontaminated math competitions.

Yang Chen, Zhuolin Yang, Zihan Liu, Chankyu Lee, Peng Xu, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. 2025. Acereason-nemotron: Advancing math and code reasoning through reinforcement learning. *CoRR*, abs/2505.16400.

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Wei Fu, Jiaxuan Gao, Xujie Shen, Chen Zhu, Zhiyu Mei, Chuyi He, Shusheng Xu, Guo Wei, Jun Mei, Jiashu Wang, Tongkai Yang, Binhang Yuan, and Yi Wu. 2025. Areal: A large-scale asynchronous reinforcement learning system for language reasoning. *CoRR*, abs/2505.24298.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, and 175 others. 2025a. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nat.*, 645(8081):633–638.

Yongxin Guo, Wenbo Deng, Zhenglin Cheng, and Xiaoying Tang. 2025b. G^2rpo-a: Guided group relative policy optimization with adaptive guidance. *CoRR*, abs/2508.13023.

Bingxiang He, Zekai Qu, Zeyuan Liu, Yinghao Chen, Yuxin Zuo, Cheng Qian, Kaiyan Zhang, Weize Chen, Chaojun Xiao, Ganqu Cui, Ning Ding, and Zhiyuan Liu. 2025a. Justrl: Scaling a 1.5b LLM with a simple RL recipe. *CoRR*, abs/2512.16649.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. 2024. Olympiadbench: A challenging benchmark for promoting AGI with olympiad-level bilingual multimodal scientific problems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 3828–3850. Association for Computational Linguistics.

Jujie He, Jiacai Liu, Chris Yuhao Liu, Rui Yan, Chaojie Wang, Peng Cheng, Xiaoyu Zhang, Fuxiang Zhang, Jiacheng Xu, Wei Shen, Siyuan Li, Liang Zeng, Tianwen Wei, Cheng Cheng, Bo An, Yang Liu, and Yahui Zhou. 2025b. Skywork open reasoner 1 technical report. *CoRR*, abs/2505.22312.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the MATH dataset. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*.

Qihan Huang, Long Chan, Jinlong Liu, Wanggui He, Hao Jiang, Mingli Song, Jingyuan Chen, Chang Yao, and Jie Song. 2025a. Boosting MLLM reasoning with text-debiased hint-grpo. *CoRR*, abs/2503.23905.

Zeyu Huang, Tianhao Cheng, Zihan Qiu, Zili Wang, Yinghui Xu, Edoardo M. Ponti, and Ivan Titov. 2025b. Blending supervised and reinforcement fine-tuning with prefix sampling. *CoRR*, abs/2507.01679.

Jia Li, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Huang, Kashif Rasul, Longhui Yu, Albert Q Jiang, Ziju Shen, and 1 others. 2024. Numinamath: The largest public dataset in ai4maths with 860k pairs of competition math problems and solutions. *Hugging Face repository*, 13:9.

Jiazheng Li, Hong Lu, Kaiyue Wen, Zaiwen Yang, Jiaxuan Gao, Hongzhou Lin, Yi Wu, and Jingzhao Zhang. 2025. Questa: Expanding reasoning capacity in llms via question augmentation. *CoRR*, abs/2507.13266.

Baohao Liao, Hanze Dong, Xinxing Xu, Christof Monz, and Jiang Bian. 2026. Self-hinting language models enhance reinforcement learning. *Preprint*, arXiv:2602.03143.

Mingyang Liu, Gabriele Farina, and Asuman E. Ozdaglar. 2025a. UFT: unifying supervised and reinforcement fine-tuning. *CoRR*, abs/2505.16984.

Shudong Liu, Hongwei Liu, Junnan Liu, Linchen Xiao, Songyang Gao, Chengqi Lyu, Yuzhe Gu, Wenwei Zhang, Derek F. Wong, Songyang Zhang, and Kai Chen. 2025b. Compassverifier: A unified and robust verifier for llms evaluation and outcome reward. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, EMNLP*

2025, Suzhou, China, November 4-9, 2025, pages 33466–33494. Association for Computational Linguistics.

Ziru Liu, Cheng Gong, Xinyu Fu, Yaofang Liu, Ran Chen, Shoubo Hu, Suiyun Zhang, Rui Liu, Qingfu Zhang, and Dandan Tu. 2025c. GHPO: adaptive guidance for stable and efficient LLM reinforcement learning. *CoRR*, abs/2507.10628.

Ivan Moshkov, Darragh Hanley, Ivan Sorokin, Shubham Toshniwal, Christof Henkel, Benedikt Schifferer, Wei Du, and Igor Gitman. 2025. AIMO-2 winning solution: Building state-of-the-art mathematical reasoning models with openmathreasoning dataset. *CoRR*, abs/2504.16891.

Vaskar Nath, Elaine Lau, Anisha Gunjal, Manasi Sharma, Nikhil Baharte, and Sean Hendryx. 2025. Adaptive guidance accelerates reinforcement learning of reasoning models. *CoRR*, abs/2506.13923.

Yuxiao Qu, Amrith Setlur, Virginia Smith, Ruslan Salakhutdinov, and Aviral Kumar. 2026. Pope: Learning to reason on hard problems via privileged on-policy exploration. *Preprint*, arXiv:2601.18779.

Yuxiao Qu, Anikait Singh, Yoonho Lee, Amrith Setlur, Ruslan Salakhutdinov, Chelsea Finn, and Aviral Kumar. 2025. RLAD: training llms to discover abstractions for solving reasoning problems. *CoRR*, abs/2510.02263.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *Preprint*, arXiv:2402.03300.

Kimi Team. 2026. Kimi K2.5: visual agentic intelligence. *CoRR*, abs/2602.02276.

Qwen Team. 2025. Qwen3 technical report. *CoRR*, abs/2505.09388.

Haifeng Wang, Hua Wu, Tian Wu, Yu Sun, Jing Liu, Dianhai Yu, Yanjun Ma, Jingzhou He, Zhongjun He, Dou Hong, Qiwen Liu, Shuohuan Wang, Junyuan Shang, Zhenyu Zhang, Yuchen Ding, Jinle Zeng, Jiabin Yang, Liang Shen, Ruibiao Chen, and 419 others. 2026. Ernie 5.0 technical report. *Preprint*, arXiv:2602.04705.

Xinyi Wang, Jinyi Han, Zishang Jiang, Tingyun Li, Jiaqing Liang, Sihang Jiang, Zhaoqian Dai, Shuguang Ma, Fei Yu, and Yanghua Xiao. 2025. HINT: helping ineffective rollouts navigate towards effectiveness. *CoRR*, abs/2510.09388.

Liang Wen, Yunke Cai, Fenrui Xiao, Xin He, Qi An, Zhenyu Duan, Yimin Du, Junchen Liu, Lifu Tang, Xiaowei Lv, Haosheng Zou, Yongchao Deng, Shousheng Jia, and Xiangzheng Zhang. 2025. Light-r1: Curriculum sft, DPO and RL for long COT from scratch and beyond. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 6: Industry Track), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pages 318–327. Association for Computational Linguistics.

Jinyang Wu, Chonghua Liao, Mingkuan Feng, Shuai Zhang, Zhengqi Wen, Haoran Luo, Ling Yang, Huazhe Xu, and Jianhua Tao. 2025. Templaterl: Structured template-guided reinforcement learning for llm reasoning. *Preprint*, arXiv:2505.15692.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, and 16 others. 2025. DAPO: an open-source LLM reinforcement learning system at scale. *CoRR*, abs/2503.14476.

Feng Zhang, Zezhong Tan, Xinhong Ma, Ziqiang Dong, Xi Leng, Jianfei Zhao, Xin Sun, and Yang Yang. 2025a. Adhint: Adaptive hints with difficulty priors for reinforcement learning. *CoRR*, abs/2512.13095.

Kaiyi Zhang, Ang Lv, Jinpeng Li, Yongbo Wang, Feng Wang, Haoyuan Hu, and Rui Yan. 2025b. Stephint: Multi-level stepwise hints enhance reinforcement learning to reason. *CoRR*, abs/2507.02841.

Xichen Zhang, Sitong Wu, Yinghao Zhu, Haoru Tan, Shaozuo Yu, Ziyi He, and Jiaya Jia. 2025c. Scaf-grpo: Scaffolded group relative policy optimization for enhancing LLM reasoning. *CoRR*, abs/2510.19807.

Xuechen Zhang, Zijian Huang, Yingcong Li, Chenshun Ni, Jiasi Chen, and Samet Oymak. 2025d. BREAD: branched rollouts from expert anchors bridge SFT & RL for reasoning. *CoRR*, abs/2506.17211.

Yifan Zhang and Team Math-AI. 2024. American invitational mathematics examination (aime) 2024.

Yifan Zhang and Team Math-AI. 2025. American invitational mathematics examination (aime) 2025.

A Visualization of the Critical-segment Effect.

To further visualize the critical-segment effect, we conduct a controlled prefix-ratio study on the QuestA dataset and randomly choose 50 training instances for visualization. For each instance, we take the reference solution and append only the first $r\%$ prefix (with r varying from 0 to 90) to the prompt followed by `## Hint`, while keeping all other decoding and evaluation settings fixed. Figure 8 shows that, for most instances, accuracy does not increase linearly as the injected prefix becomes longer. Instead, performance typically remains flat in low-ratio regions and then exhibits a distinct jump once a key segment is included, followed by diminishing gains. This pattern supports our view: effective guidance depends on whether critical knowledge is covered, rather than on monotonically increasing hint length.

B Prompts

B.1 Example Augmented Prompt

This section presents a concrete data example of our augmented prompt format; see Figure 9.

B.2 KP Curation Details and Prompts

This section provides additional details for the KP curation pipeline introduced in Section 3.1. The pipeline consists of three stages. First, for each problem, we use an existing verified correct solution as the reasoning trajectory from which knowledge should be extracted. Second, given the problem and its correct solution, we prompt `DeepSeek-R1` to extract only the indispensable mathematical principles required to solve the problem, yielding an initial candidate KP set $\mathcal{K} = \{k_1, k_2, \ldots, k_n\}$. Third, to prevent information leakage, we verify each KP using `DeepSeek-R1` as an automated reviewer, and manually revise failed cases to ensure that all retained KPs are generalizable and not instance-bound.

The prompts for extracting raw knowledge points and verifying leakage are shown in Figure 10 and Figure 11, respectively.

C Effect of Tolerance Threshold

To set the tolerance threshold in CBRS, we compare $\delta \in \{0/32, 1/32, 2/32\}$. Figure 2a shows that with $\delta = 0/32$, the intersection of near-optimal candidates across runs is often too small, making selection brittle; with $\delta = 2/32$, the overlap rises to around 60%, but the selected KP set becomes much larger. As reported in Table 4, $\delta = 1/32$ achieves the best overall result (highest average accuracy, 63.90) while keeping the KP count moderate (2.57), providing a strong balance between performance and compactness.

D Offline KP Selection Results on 7B

To complement the 1.5B offline analysis in Figure 4, we also report the corresponding results on DeepSeek-R1-Distill-Qwen-7B in Figure 12. The 7B backbone shows the same overall trend: adding KP guidance improves offline accuracy over the no-KP setting, while the selected-KP setting reduces the number of injected KPs compared with using the full raw KP set.

E Entropy Annealing Analysis

To accelerate convergence under a limited training budget, we apply entropy annealing by adjusting the clip upper bound during training. As shown in Figure 13, we illustrate its effect using the KnowRL-Nemotron-1.5B training run. With `clip_high`=0.28, entropy first increases in the early stage, encouraging exploration, and then begins to decrease around step 2,590 as the model searches for more optimal reasoning paths. To further accelerate convergence, we reduce `clip_high` from 0.28 to 0.26 after step 2,590. This tighter clipping regime induces a faster entropy drop, encouraging the policy to shift earlier from exploration to exploitation, which helps the model reach stronger performance within fewer optimization steps.

To isolate the contribution of this strategy, we compare against a control setting that keeps `clip_high`=0.28 throughout training and runs to the same 2,960-step budget. Table 6 further reports the detailed results on eight evaluation benchmarks, where the entropy-annealed setting achieves better overall scores than both the non-annealed variant and JustRL.

F Benchmark Details

Table 5 summarizes the eight mathematical reasoning benchmarks used for offline evaluation, and offline selection analysis. These benchmarks contain 1,374 problems in total and span recent mathematical competitions, curated MATH problems, and olympiad-level reasoning tasks. All KP construction and selection procedures are performed

δ	AIME24	AIME25	BRUMO25	HMMT Feb 25	AMC23	CMIMC25	MATH-500	Olympiad Bench	Avg.	Avg. #KP
0/32	63.13	49.90	64.06	34.48	91.60	33.40	92.55	73.95	62.88	1.19
1/32	64.44	50.57	65.03	35.77	91.71	36.70	92.90	74.11	63.90	2.57
2/32	64.05	50.30	64.80	35.20	91.33	35.90	92.85	73.70	63.52	3.45

Table 4: Effect of tolerance threshold δ on offline performance and KP compactness. $\delta = 1/32$ provides the best balance between average accuracy and average number of selected KPs.

via offline evaluation before RL training, which improves reproducibility and avoids online teacher interaction during RL.

G Out-of-domain Generalization for 7B Models

Table 7 reports the corresponding out-of-domain evaluation for the 7B setting, covering code, general, and scientific reasoning benchmarks.

H More Analysis

H.1 Training Cost Analysis

To make the efficiency comparison clearer, we report an explicit training-budget comparison across hint-based methods in Table 8. Following the common approximation Token Budget $\approx$ Training Steps $\times$ Train Batch Size $\times$ Rollout N $\times$ Max Context Length, KnowRL uses a smaller RL token budget than QuestA, but a larger budget than JustRL-Nemotron because of its longer 24k context length. Therefore, our claim should not be interpreted as KnowRL always requiring the fewest raw RL training tokens among all baselines. Rather, the efficiency advantage of KnowRL lies in converting guidance generation into a one-time offline curation stage and avoiding repeated teacher interaction during RL training.

We also report the offline KP construction cost. In our implementation, KP construction and selection take approximately 50 hours on 8 nodes with 8 H800 GPUs each. This includes leakage verification, offline evaluation for KP selection, and CSS-based subset search. Once constructed, the curated KPs are reused throughout RL training and do not require additional teacher-model inference during online optimization. This is important because many hint-based RL methods that interact with a teacher model during training require either deploying an additional teacher model or calling an external API throughout RL. Such teacher interaction introduces extra computation, latency, and scheduling overhead. Teacher-model rollout itself introduces a non-negligible additional cost, roughly on the order of one fifth of the student-model rollout time, and deployment of the teacher model further increases the resource burden. By contrast, KnowRL shifts this cost to an offline, reusable preprocessing stage.

H.2 Improvements on Training Data

To further characterize KnowRL's effect, we analyze the per-query correct-count distribution (out of 8 samples) over the training set across three conditions, as shown in Figure 14.

The backbone suffers severely from reward sparsity: 41.21% of queries receive zero correct answers and only 1.35% are solved consistently, yielding a mean accuracy of 22.40%. KnowRL training alone (w/o KPs at inference) collapses the zero-correct fraction to 13.00% and raises the all-correct bucket to 34.28% (+32.93pp), lifting average accuracy to 64.30%. This confirms that KP-guided training genuinely internalizes structured reasoning rather than producing hint-conditioned shortcuts. Adding KP hints at inference further concentrates mass at the rightmost bucket (51.07%), with mid-range counts (1–6) each shrinking by 2–3 percentage points, consistent with the critical-segment effect: once minimal sufficient knowledge is made explicit, the model resolves partial successes into consistent correctness. Average accuracy reaches 77.04% under this condition.

H.3 KP Extractor Sensitivity Details

Table 9 compares offline KP selection results under different KP extractors and data selection strategies. Using Qwen3-8B as the extractor yields a CSS-selected average accuracy close to DeepSeek-R1 (63.76 vs. 63.90), while preserving the same overall trend that compact selection strategies outperform raw all-KP injection. This supports the conclusion that KnowRL's KP curation pipeline can remain effective with a lower-cost extractor,

Benchmark	# Problems	Source	Answer Format	Description
AIME24 (Zhang and Math-AI, 2024)	30	AIME 2024	Integer answer	Problems from the 2024 American Invitational Mathematics Examination, covering algebra, geometry, number theory, combinatorics, and probability.
AIME25 (Zhang and Math-AI, 2025)	30	AIME 2025	Integer answer	Problems from the 2025 American Invitational Mathematics Examination, used to evaluate recent contest-level mathematical reasoning.
BRUMO25 (Balunović et al., 2025)	30	MathArena / BRUMO 2025	Short answer	Recent competition problems from BRUMO 2025, extracted, converted to LaTeX, and verified by MathArena.
HMMT-Feb-25 (Balunović et al., 2025)	30	MathArena / HMMT February 2025	Short answer	Problems from the February 2025 Harvard–MIT Mathematics Tournament, covering contest-level mathematical reasoning.
AMC23 (Li et al., 2024)	40	AMC 2023	Integer / short answer	Problems from the 2023 American Mathematics Competitions, commonly used as a compact benchmark for high-school-level mathematical reasoning.
CMIMC25 (Balunović et al., 2025)	40	MathArena / CMIMC 2025	Short answer	Problems from the 2025 Carnegie Mellon Informatics and Mathematics Competition, included in MathArena's recent-competition evaluation suite.
MATH-500 (Hendrycks et al., 2021)	500	MATH subset	Short answer	A curated subset of 500 problems from the MATH benchmark, covering multiple subjects such as algebra, geometry, number theory, probability, and precalculus.
Olympiad-Bench (He et al., 2024)	674	OlympiadBench subset	Open-ended / short answer	A processed subset of OlympiadBench used for olympiad-level mathematical reasoning evaluation.
Total	**1,374**	–	–	–

Table 5: Summary of the eight mathematical reasoning benchmarks used for KP annotation, offline analysis, and evaluation. The Olympiad-Bench entry corresponds to the processed 674-problem subset used in our evaluation rather than the full OlympiadBench benchmark.

Model	AIME24	AIME25	BRUMO25	HMMT25	AMC23	CMIMC25	MATH	OlyBench	Avg.
KnowRL-Nemotron-1.5B	69.79	64.69	69.45	41.04	95.55	44.17	95.70	80.23	70.08
w/o entropy annealing	68.65	62.19	67.40	39.27	95.94	42.81	94.67	77.95	68.61
JustRL	69.69	62.92	66.88	40.63	96.02	41.72	94.15	76.59	68.58

Table 6: Ablation of entropy annealing on KnowRL-Nemotron-1.5B across eight evaluation benchmarks.

provided that the same leakage filtering and selection stages are applied.

H.4 All-KP versus CSS Training Dynamics

We compare two 1.5B RL training variants that differ only in the hint data used during training: one injects the full raw KP set for each hinted problem, and the other uses CSS-selected KPs. Figure 15 summarizes the held-out benchmark perfrmance. The two settings are close on training accuracy, indicating that full-KP hints can still make training problems easier to solve. However, their test performance differs substantially. Under full-KP training, no-KP evaluation on held-out benchmarks tends to stagnate or decline, whereas CSS-selected training continues to improve. We attribute this gap to overfitting to raw hint annotations: with all KPs, the model learns to condition on both useful and low-utility KPs, while CSS exposes it to more compact and interaction-aware guidance that transfers better when no external KP hints are available.

We further analyze the source of the generalization gap between different KP training strategies through a case study. We observe that after KP-based training, models tend to proactively generate "knowledge points" to assist problem solving even when no explicit KP hint is provided at inference time. We refer to these self-generated items as *fictional KPs*: they are plausible-looking hints produced from surface features of the problem and used to guide the subsequent reasoning process.

However, the quality of fictional KPs depends strongly on the KP distribution seen during training. For models trained with the full raw KP set, exposure to both useful and useless KPs may encourage the model to solve problems by conditioning on all available knowledge points, rather than learning the underlying minimal-sufficient reasoning structure. As a result, during no-KP inference, the model may generate fictional KPs that contain irrelevant or even misleading content, which can further steer the reasoning process toward an incorrect solution

Model	LiveCodeBench pass@1	LiveCodeBench pass@5	MMLU	ARC-Challenge	GPQA-Extended	GPQA-Main
DeepSeek-R1-Distill-Qwen-7B	49.44	65.31	61.89	79.69	48.72	50.45
Light-R1-7B-DS	51.32	66.16	59.51	76.45	49.45	50.22
AReaL-boba-RL-7B	46.27	63.32	63.39	80.80	48.17	**53.79**
SkyWork-OR1-7B	**57.95**	**69.48**	61.46	80.12	52.01	52.23
KnowRL-DS-Distill-Qwen-7B	56.15	68.08	**63.42**	**81.23**	**53.11**	52.90

Table 7: Out-of-domain generalization of KnowRL-DS-Distill-Qwen-7B across code, general, and scientific reasoning benchmarks.

Model	Training Steps	Train Batch Size	Rollout N	Max Context Length	RL Token Budget (approx.)
QuestA	2,000	128	16	32k	2.6×10^8k
JustRL-Nemotron	3,440	256	8	16k	1.1×10^8k
KnowRL-Nemotron-1.5B	2,960	256	8	24k	1.5×10^8k

Table 8: Approximate RL token budgets for hint-based methods. The budget is estimated as Training Steps $\times$ Train Batch Size $\times$ Rollout N $\times$ Max Context Length.

and degrade generalization.

In contrast, CSS-selected KP training filters out redundant or noisy KPs while preserving effective guidance signals. Therefore, although the model may still verbalize intermediate knowledge during inference, these generated knowledge points are more likely to correspond to the genuine concepts and key reasoning steps required for solving the problem, rather than merely imitating the surface form of KP annotations.

Figures 16 and 17 contrast two no-KP inference responses to the same problem. Figure 16 shows the response from the model trained with the full raw KP set: it generates plausible but solution-irrelevant fictional KPs and follows an incorrect reasoning path. Figure 17 shows the response from the model trained with CSS-selected KPs: it verbalizes knowledge that matches the necessary concepts and key reasoning steps, leading to the correct answer. This case shows that filtering the hint set is important not only for improving compactness, but also for reducing overfitting to noisy or fictional KP patterns, thereby improving generalization under no-KP inference.

H.5 SFT Baseline Details

To address whether the gains of KnowRL can be reproduced by directly distilling DeepSeek-R1-generated correct solutions, we construct two SFT baselines on the same 1.5B backbone: one trained with KP-augmented SFT data and one trained without KP annotations. Both baselines use DeepSeek-R1-generated correct solutions as supervised training data.

All models are evaluated under the same 32k decoding setting with temperature 0.7 and top-p 0.9, except that MATH-500 and Olympiad-Bench are evaluated with $n = 8$ following our evaluation protocol. The SFT baselines are trained for 5 epochs with batch size 256, learning rate 1×10^{-4}, cosine learning-rate scheduling, and weight decay 0.01.

Under this setting, SFT from DeepSeek-R1-generated correct solutions improves only marginally over the initial Nemotron-1.5B model and remains far below KnowRL-Nemotron-1.5B without KPs at inference. Moreover, adding KPs to SFT does not lead to a meaningful improvement over SFT without KPs. This indicates that the gains of KnowRL do not come simply from exposing the model to R1-generated knowledge or solution content in any form. Instead, the improvement depends on how KPs are used in the RL process: compact, selected KPs help unlock reward-yielding trajectories under sparse rewards, while the RL objective further reinforces successful reasoning behaviors.

H.6 Selection Strategy Checkpoint Results

This section complements the selection-strategy ablation in the main text with checkpoint-level generalization results under matched training budgets. As discussed in the main text, Figure 5 shows that CSS leads to steadier optimization dynamics and stronger training accuracy. Table 10 further confirms that this advantage transfers to evaluation benchmarks at both intermediate checkpoints, improving from 64.68 to 65.00 at step 400 and from

Selection Strategy	AIME24	AIME25	BRUMO25	HMMT Feb 25	AMC23	CMIMC25	MATH-500	Olympiad Bench	Avg.
DEEPSEEK-R1 EXTRACTOR									
w/o KP	58.75	48.44	61.67	30.10	90.55	30.08	92.40	71.70	60.46
All KP	60.94	49.06	61.15	32.50	89.69	32.34	92.23	70.55	61.06
Random	60.52	49.27	61.04	33.23	91.02	31.09	91.65	71.88	61.21
Max-Score	62.60	49.79	64.27	34.79	90.94	32.97	92.53	73.89	62.72
S-LOO	62.71	49.27	63.85	33.54	91.72	33.52	92.90	73.70	62.65
T-LOO	62.08	49.27	64.17	33.65	91.25	33.67	92.40	73.46	62.49
CBRS	63.02	49.90	64.17	34.79	91.56	33.59	92.65	73.89	62.95
CSS	64.48	50.52	65.00	35.73	91.72	36.72	92.90	74.11	63.90
QWEN3-8B EXTRACTOR									
w/o KP	57.60	50.00	60.83	31.67	89.45	29.06	92.75	71.42	60.35
All KP	60.21	47.50	62.40	31.25	91.72	31.25	91.98	70.22	60.81
Max-Score	63.75	48.02	65.52	33.44	92.19	31.80	92.80	74.09	62.70
S-LOO	61.15	50.83	62.40	35.10	90.70	34.61	92.55	73.44	62.60
T-LOO	63.44	47.92	62.81	35.00	92.81	32.66	92.08	73.72	62.55
CBRS	61.46	51.46	65.63	33.13	90.63	34.77	92.95	73.66	62.96
CSS	65.83	49.17	63.75	36.04	92.66	36.02	92.33	74.31	63.76

Table 9: Offline KP selection performance using different KP extractors and data selection strategies. The DeepSeek-R1 extractor block follows the strategy comparison in Table 1.

65.72 to 66.46 at step 900.

Model	AIME24	AIME25	BRUMO25	HMMT25	AMC23	CMIMC25	MATH	OlyBench	Avg.
CBRS $_{\text{step400}}$	64.58	56.77	63.96	34.48	93.52	35.39	93.30	75.41	64.68
CSS $_{\text{step400}}$	65.94	57.08	64.38	35.31	92.73	36.25	92.85	75.46	65.00
CBRS $_{\text{step900}}$	65.42	58.85	64.79	37.08	94.14	35.70	94.00	75.78	65.72
CSS $_{\text{step900}}$	67.19	59.17	65.52	39.06	93.91	37.03	93.77	76.04	66.46

Table 10: Comparison between CBRS- and CSS-selected training data under matched training budgets (steps 400 and 900) across eight reasoning benchmarks.

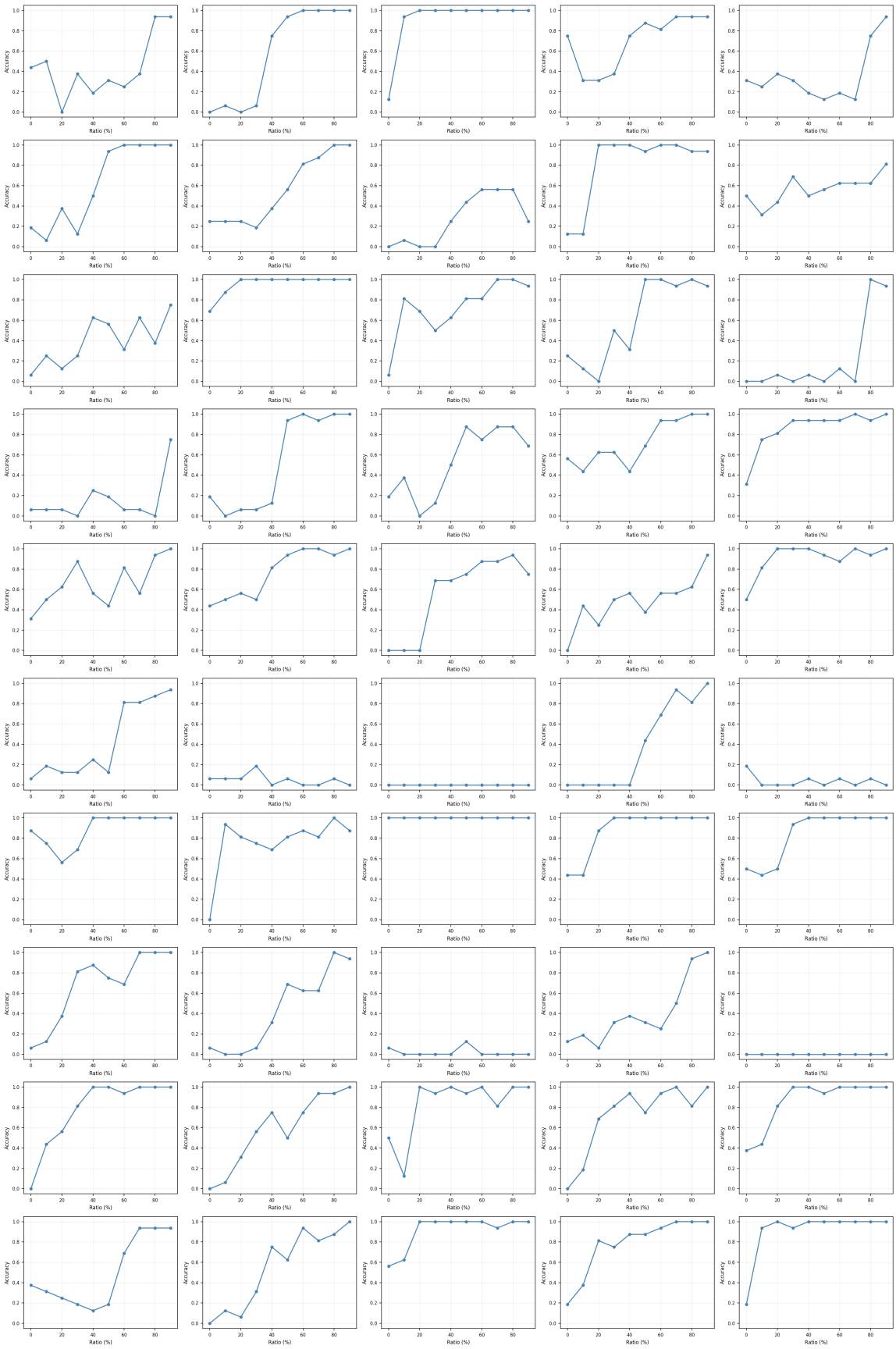


Figure 8: Visualization of the critical-segment effect across prefix ratios on 50 training instances. Each subplot reports one instance, showing that performance often changes abruptly once the prefix covers a critical segment rather than increasing smoothly with hint length.

Augmented Prompt

Jackson's paintbrush makes a narrow strip with a width of 6.5 millimeters. Jackson has enough paint to make a strip 25 meters long. How many square centimeters of paper could Jackson cover with paint?

Hint

1. **Knowledge Point**: Unit conversion between metric length units follows powers of 10 (millimeters, centimeters, meters, etc.).
Key Considerations: Moving to a larger unit requires division; moving to a smaller unit requires multiplication. The conversion factors are: 10 mm = 1 cm, 100 cm = 1 m.
2. **Knowledge Point**: When calculating area, all linear measurements must be converted to the same unit before computation.
Key Considerations: The choice of unit should match the desired unit of the final answer; converting before multiplication avoids errors in dimensional analysis.
3. **Knowledge Point**: A narrow strip of constant width covering a certain length can be modeled as a rectangle.
Key Considerations: This applies when the strip has uniform width throughout its entire length; the area represents the total surface covered.

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

Figure 9: Example augmented prompt with a partial-solution hint

Extract Raw KPs

You will be given:
1. A mathematics problem.
2. A correct solution to the problem.

Your task is to extract the essential mathematical knowledge required to solve the problem from the correct solution.
Requirements:
- List only the core knowledge points that are indispensable for solving the problem.
- Each knowledge point should be concise, general, and mathematically fundamental (not problem-specific tricks), yet crucial to the solution.
- Do NOT explain the full solution or reproduce the reasoning steps.
- Do NOT restate or paraphrase the problem.
- For each knowledge point, explicitly describe important details, conditions, or caveats that must be considered when applying it.

Format:
- Use a numbered list.
- Each item should consist of:
(a) the knowledge point, and
(b) the key considerations when applying it.

[Problem]
`{question}`
[Correct Solution]
`{solution}`
[Key Knowledge Points]

Figure 10: Prompt used for extracting raw knowledge points.

Leakage Verification

You are an expert reviewer for mathematical reasoning datasets.
You will be given:
1. A mathematics problem.
2. A candidate knowledge description.

Your task is to determine whether the knowledge description is STRONGLY COUPLED to the problem.
Strong Coupling means the knowledge description:
- Contains specific numerical values, constants, or quantities appearing in the problem or its solution (beyond common constants like π).
- Mentions specific object names, variable names, or configurations unique to the problem.
- Includes intermediate numerical results or the final answer.
- Encodes the problem's structure implicitly (e.g., "in this case", "here", "for this triangle", "for this sequence").
If ANY of the above holds, the knowledge description IS strongly coupled.
Respond STRICTLY in JSON format:
{{
"strongly_coupled": true false,
"reason": "<brief explanation>"
}}

[Problem]
`{question}`
[Knowledge Description]
`{knowledge}`.

Figure 11: Prompt used for leakage verification with an augmented hint.

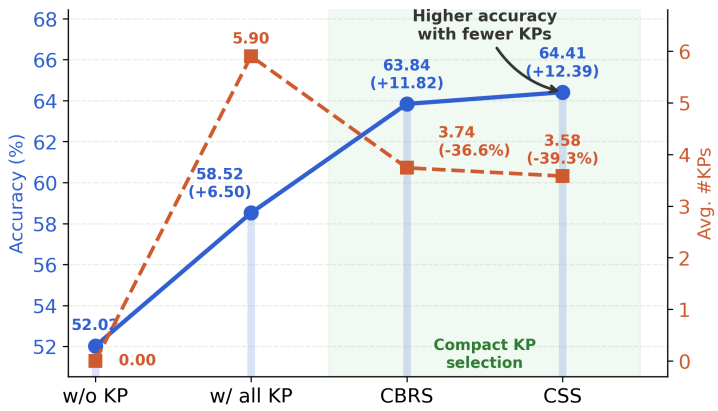


Figure 12: Offline evaluation on the QuestA dataset with different KP selection strategies for DeepSeek-R1-Distill-Qwen-7B, showing the trade-off between accuracy and the average number of selected KPs.

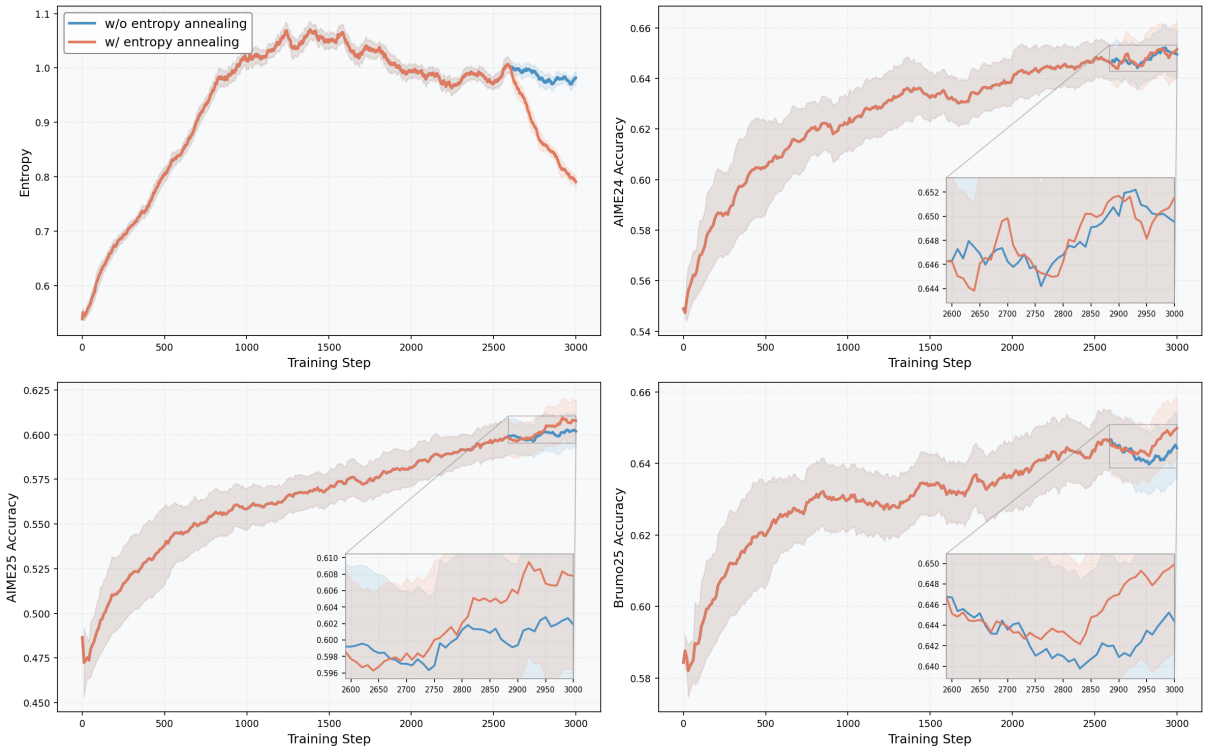


Figure 13: Comparison with and without entropy annealing. We report the entropy trajectory and the corresponding performance on different validation benchmarks under the 24k setting during training. Entropy annealing yields faster entropy reduction and consistently better validation performance.

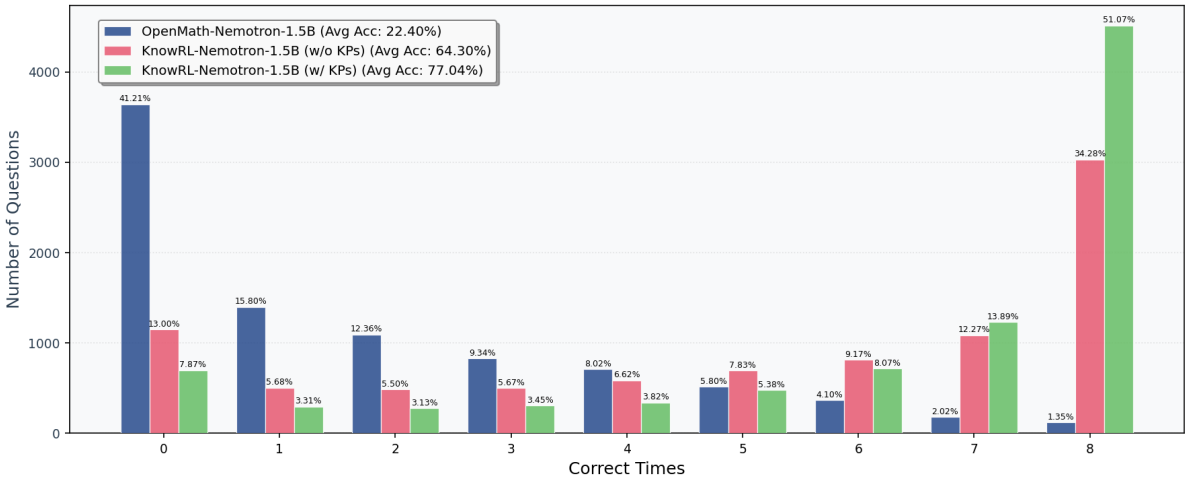


Figure 14: Distribution of per-query correct counts on the training set for OpenMath-Nemotron-1.5B, and KnowRL-Nemotron-1.5B model under two offline evaluation settings: without KP hints and with KP hints at inference.

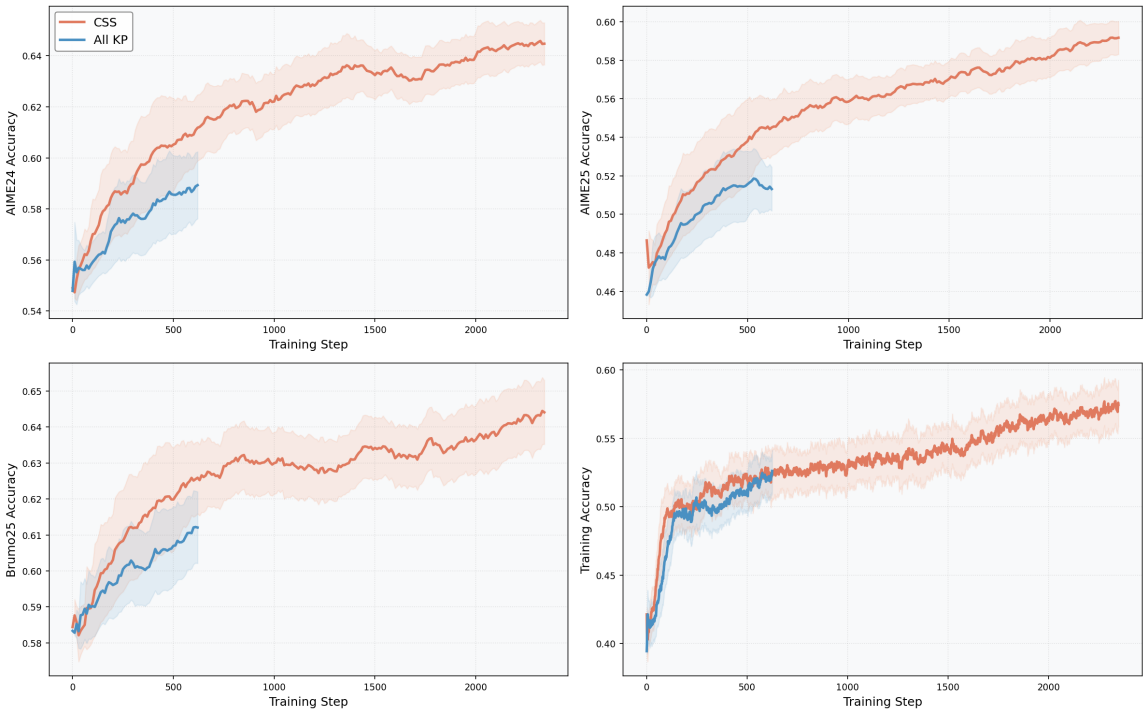


Figure 15: Comparison between full-KP hints and CSS-selected KP hints. CSS-selected hints provide stronger held-out performance than injecting all raw KPs, supporting the need for compact, interaction-aware KP selection.

No-KP response from full-KP training

Problem. Rectangles $ABCD$ and $EFGH$ are drawn such that D, E, C, F are collinear. Also, A, D, H, G all lie on a circle. If $BC = 16$,$AB = 107$,$FG = 17$, and $EF = 184$, what is the length of CE?

Full-KP-trained model. <think>
Okay, let me try to figure out this problem. So, there are two rectangles, $ABCD$ and $EFGH$. The points D, E, C, F are collinear, and points A, D, H, G lie on a circle. We need to find the length of CE. The given lengths are $BC = 16$, $AB = 107$, $FG = 17$, and $EF = 184$.
...
Let me place D at $(0, 0)$ and C at $(107, 0)$. Then $A = (107, 16)$. Let $E = (e, 0)$ and $F = (f, 0)$, with $EF = 184$. Since $EFGH$ is a rectangle, take $H = (e, 17)$ and $G = (f, 17)$.
...
Alternatively, since A, D, H, G are concyclic, maybe I can use the power of a point.
Power of point E with respect to the circle gives

$$ED \cdot EC = EH \cdot EG.$$

Here, E lies on the line D, E, C, F, so the line through E intersects the circle at D and C. Also, H and G are points on the circle related to the rectangle, so another intersection product is $EH \cdot EG$.
...
Thus,

$$e \approx \frac{107 + 155}{2} = 131.$$

Then

$$CE = e - 107 = 131 - 107 = 24.$$

...
Final Answer

$$\boxed{24}$$

</think>
<answer>
...
Thus,

$$\boxed{24}$$

</answer>

Figure 16: No-KP inference response from the model trained with the full raw KP set. The model self-generates plausible but misleading fictional KPs, treats the matching events as independent Bernoulli trials, and reaches an incorrect answer.

No-KP response from CSS-selected KP training

Problem. Rectangles $ABCD$ and $EFGH$ are drawn such that D, E, C, F are collinear. Also, A, D, H, G all lie on a circle. If $BC = 16$,$AB = 107$,$FG = 17$, and $EF = 184$, what is the length of CE?

CSS-trained model. <think>
Okay, let's try to tackle this geometry problem. Hmm, we have two rectangles, $ABCD$ and $EFGH$.
...
Let me consider that the line D, E, C, F is the x-axis, with D at $(0, 0)$, C at $(107, 0)$, E at $(e, 0)$, and F at $(f, 0)$, with $0 < e < 107 < f$.
Rectangle $ABCD$: $AB = 107$, $BC = 16$. Since $DC = AB = 107$ and $BC = AD = 16$, I can place

$$A = (0, -16), \quad B = (107, -16), \quad C = (107, 0), \quad D = (0, 0).$$

...
For rectangle $EFGH$, since D, E, C, F are collinear and $EF = 184$, we can write

$$E = (e, 0), \qquad F = (e + 184, 0).$$

Also, since $FG = 17$ and $EF \perp FG$,

$$G = (e + 184, 17), \qquad H = (e, 17).$$

Now, the key condition is that A, D, H, G are concyclic.
...
Solving,

$$e = \frac{-184 \pm \sqrt{184^2 + 4 \cdot 561}}{2}.$$

...
Finally,

$$CE = 107 - e = 107 - 3 = 104.$$

Final Answer

$$\boxed{104}$$

</think>
<answer>
Because A, D, H, G are concyclic, use

$$x^2 + y^2 + Dx + Ey + F = 0.$$

...

$$\boxed{104}$$

</answer>

Figure 17: No-KP inference response from the model trained with CSS-selected KPs. The generated intermediate knowledge aligns with the key conditional-probability and combinatorial structure, guiding the model to the correct answer.