SURGICAL POST-TRAINING: IMPROVING REASONING WHILE MITIGATING FORGETTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Injecting new reasoning knowledge into Large Language Models (LLMs) via post-training often induces catastrophic forgetting. Recent studies emphasize the importance of on-policy data in mitigating forgetting, but suggest that explicit KL regularization does not explain the observed retention benefits. We study the implicit regularization induced by reference-relative reward objectives: gradient analysis reveals adaptive attenuation of updates, while controlled comparisons on the same rectified data demonstrate improved capability retention. This motivates our Surgical Post-Training (SPOT), a post-training framework designed to optimize reasoning efficiently while preserving prior knowledge. SPOT consists of (1) a data rectification pipeline employing an Oracle to surgically correct erroneous steps via minimal edits, generating proximal on-policy data; and (2) a reward-based binary cross-entropy objective that learns from both corrected and erroneous responses. Empirically, with only 4k rectified math pairs, SPOT improves Qwen3-8B's accuracy by 6.3 percentage points on average across 9 in-domain and out-of-domain tasks, requiring merely 16 minutes of model training on 8× H800 GPUs. We further validate SPOT through Connect4 post-training, where it improves both game and mathematical reasoning while retaining instruction-following capabilities. Moreover, SPOT provides a superior initialization for subsequent reinforcement learning, enabling further gains in reasoning performance.

1 INTRODUCTION

While pretraining builds the foundation of Large Language Models (LLMs) (Achiam et al., 2023; Comanici et al., 2025), post-training remains critical for eliciting specific competencies, such as mathematical reasoning (Shao et al., 2024), code generation (Anthropic, 2024), and human alignment (Ouyang et al., 2022; Zhou et al., 2023). Post-training for reasoning primarily follows two paradigms: Supervised Fine-Tuning (SFT) and Reinforcement Learning (RL). While SFT benefits from strong supervision, it can induce catastrophic forgetting of prior capabilities outside the training domain when injecting new reasoning capabilities (Chu et al., 2025; Huan et al., 2025). On-policy RL methods like Group Relative Policy Optimization (GRPO) (Shao et al., 2024) can mitigate forgetting and excel on tasks with verifiable outcomes, yet they rely on the base model's ability to sample correct paths, making them less suited for injecting new knowledge that expands fundamental reasoning boundaries (Yue et al., 2025).

This raises a critical question: **How can we effectively inject external reasoning knowledge into LLMs while mitigating catastrophic forgetting?** Recent studies highlight the role of *on-policy data* in mitigating forgetting by reducing distributional shifts between the training data and the model's policy (Chen et al., 2025; Shenfeld et al., 2025). Motivated by this, we introduce a data rectification pipeline designed to provide external reasoning supervision while limiting changes to the student's own responses. Specifically, we employ an Oracle (e.g., a teacher model) to surgically correct erroneous student outputs via minimal edits, yielding positive samples that remain proximal to the model's original distribution. This pipeline requires only text-based Oracle feedback, without the teacher token distributions used in token-level on-policy distillation (Agarwal et al., 2024; Lu & Thinking Machines Lab, 2025; Yang et al., 2025; Xiao et al., 2026).

However, we find that data proximity alone is insufficient to prevent forgetting. Empirically, SFT on this proximal data *still* reduces instruction-following performance, whereas Direct Preference

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.

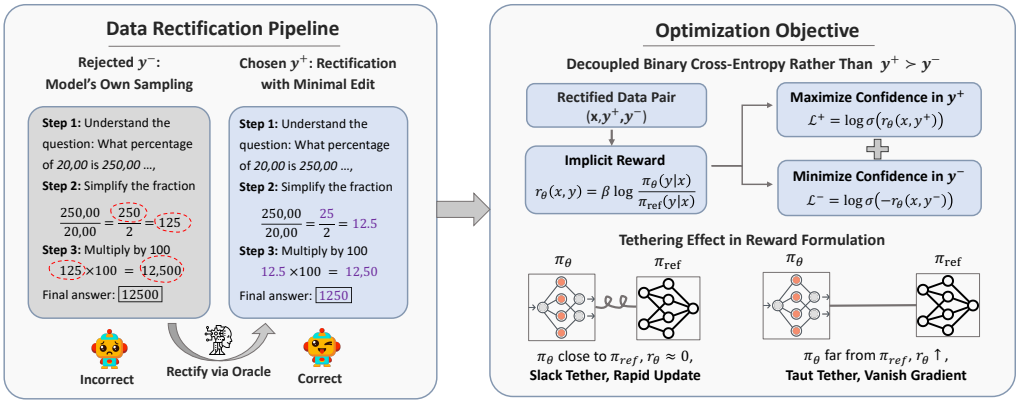


Figure 1: **Illustration of SPoT.** Left: An Oracle surgically corrects reasoning errors, yielding positive samples proximal to the model's original distribution. Top-Right: Unlike DPO's relative ranking, we leverage an explicit classification loss that proves more effective for reasoning tasks. Bottom-Right: The tethering effect inherent in our reward definition substantially mitigates catastrophic forgetting.

Optimization (DPO) (Rafailov et al., 2023) preserves it. To understand the mechanism, we employ a Reward-SFT baseline, a method integrating the DPO reward formulation into SFT. We observe that Reward-SFT also preserves this capability, suggesting that the reference-relative reward formulation contributes to retention by attenuating updates to responses already favored over the reference model.

However, DPO and Reward-SFT provide limited in-domain reasoning gains in our experiments, motivating a closer examination of their learning signals. We identify two failure modes: (1) *the "pull-up" effect* (Ren & Sutherland, 2025) inherent in positive-only training (e.g., SFT), which inadvertently raises the likelihood of erroneous responses alongside correct ones; and (2) *the inadequacy of relative ranking* in DPO for reasoning with verifiable correctness.

To address these limitations, we propose Surgical Post-Training (SPoT), a paradigm designed to optimize reasoning efficiently while preserving prior knowledge. SPoT integrates our data rectification pipeline with a reward-based binary cross-entropy objective to maintain the "tethering" effect of the reference model. Unlike the relative ranking in DPO, our objective explicitly maximizes the likelihood of the rectifications while suppressing errors, showing superior gains in reasoning.

In this paper, we make the following contributions. *Firstly*, we show that the training objective also affects retention on the same proximal data, with DPO and Reward-SFT preserving instruction-following performance better than SFT. *Secondly*, we propose SPoT, a post-training framework that mitigates catastrophic forgetting by synergizing minimal-edit data rectification with a reward-based objective; we identify the "pull-up" effect in positive-only training and the insufficiency of DPO for reasoning, and validate a binary classification objective that provides a denser signal. *Thirdly*, we show that SPoT establishes a superior initialization for subsequent reinforcement learning: compared to applying GRPO directly, initializing GRPO with SPoT checkpoints substantially unlocks higher performance ceilings on in-domain math (+7.1 percentage points) and Connect4 (+21.7 percentage points), with better general instruction-following capabilities. *Lastly*, to strictly evaluate OOD reasoning without data contamination, we leverage GAMEBoT (Lin et al., 2025) to dynamically construct a `Connect4` dataset for reliable evaluation. We additionally validate SPoT on Connect4 post-training, showing improved game and mathematical reasoning with better instruction-following retention than SFT.

2 Data Rectification Pipeline

Our SPoT framework consists of two core components: a *data rectification pipeline* and a *specialized optimization objective* (see Fig. 1). In this section, we detail the first component, which is designed to produce valid reasoning paths proximal to the model's distribution.

Standard SFT often utilizes offline datasets that diverge significantly from the model's current policy π_θ, leading to catastrophic forgetting (Chen et al., 2025). Conversely, on-policy methods rely entirely

on the model's intrinsic probability of sampling correct reasoning paths, making them inefficient for hard reasoning tasks. For example, if a model's pass rate on a specific problem is 1%, obtaining a single correct response requires roughly 100 rollouts in expectation, which is computationally expensive. Furthermore, for problems beyond the model's current capabilities, self-sampling fails to yield any training signal, making it impossible to learn from those instances (Yu et al., 2025).

To bridge this gap, we propose a *data rectification pipeline* to construct a contrastive dataset $\mathcal{D} = \{(x, y^-, y^+)\}$ where the positive response y^+ only performs necessary corrections to the erroneous logic of the negative response y^- while preserving the model's original generation style and lexical structure. An example is shown in Fig. 1. The pipeline consists of the following three stages:

Error Elicitation. We first construct a dataset of model failures. Given a question x from the source dataset, we sample a response y^- from the current policy: $y^- \sim \pi_\theta(\cdot|x)$. We evaluate y^- against the ground-truth answer. If the final answer is incorrect, we retain the pair (x, y^-) for rectification.

Oracle-Guided Surgical Rectification. We employ an Oracle (e.g., a human expert or a teacher model) to perform surgical edits. The Oracle is provided with y^- and optionally the ground truth, then instructed to only modify incorrect reasoning steps, while maintaining the original style. After filtering out any outputs with incorrect final answers, we obtain a rectified response y^+ that represents the "nearest valid neighbor" to the erroneous response y^- in the semantic space. To enhance efficiency, we adopt an *offline* setting (i.e., the Oracle is queried once per failure rather than at each training step). We provide the full prompts in Appendix C.1, and verify the framework's robustness to a weaker open-source Oracle and to prompt formulation in Appendix E.1 and Appendix E.2, respectively.

LCS Filtering. To ensure the proximity of the rectified data to the student's distribution, we apply a structural constraint based on the Longest Common Subsequence (LCS) (Wagner & Fischer, 1974). For every pair (y^-, y^+), we calculate the change ratio R_{LCS}:

$$R_{LCS}(y^-, y^+) = 1 - \frac{|\text{LCS}(y^-, y^+)|}{|y^+|}, \tag{1}$$

where LCS is computed over the tokenized responses, and $|\cdot|$ denotes the number of tokens. We filter out samples where $R_{LCS} > \gamma$. We set $\gamma = 0.6$ in the main experiments (see Appendix E.3 for the filtering trade-off). Appendix C.2 shows the change ratio distributions.

In the resulting dataset, y^- and y^+ in each sample retain substantial token overlap while introducing localized corrections. With the paired objective below, this can help focus the gradients on divergent reasoning tokens during optimization.

3 Reward-Based Objectives as Implicit Regularizers

Using the proximal dataset from Sec. 2, we demonstrate that data proximity alone is insufficient to prevent catastrophic forgetting. We analyze the learning dynamics of SFT versus reward-based methods, showing how the reward formulation acts as an intrinsic regularizer through adaptive gradient attenuation.

3.1 Empirical Observation: The Regularization Gap

We conduct controlled experiments using the English subset of DAPO-Math-17k (Yu et al., 2025). Leveraging our pipeline, we generate 4k contrastive pairs by employing Qwen3-8B as the base policy π_θ and Gemini 2.5 Pro as the Oracle for rectification. We compare the following methods, all trained on $\mathcal{D} = \{(x, y^-, y^+)\}$, and evaluate OOD generalization using IFEval (Zhou et al., 2023):
SFT+. To distinguish it from standard fine-tuning on raw off-policy data, we refer to Supervised Fine-Tuning performed on the surgically rectified positive samples y^+ as SFT+, which minimizes the same negative log-likelihood as SFT:

$$\mathcal{L}_{\text{SFT}} = -\mathbb{E}_{x,y^+\sim\mathcal{D}}[\log \pi_\theta(y^+|x)] \tag{2}$$

DPO. DPO optimizes a policy by leveraging a relative ranking objective derived from an implicit reward $r_\theta(x, y)$. It is defined as the log-ratio between the policy π_θ and the frozen reference model

π_{ref} (initialized with the same parameters as π_θ), scaled by a coefficient β:

$$r_\theta(x, y) = \beta \log \frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)} \tag{3}$$

DPO maximizes the margin between the chosen response y^+ and the rejected response y^-:

$$\mathcal{L}_{\text{DPO}} = -\mathbb{E}_{x,y^+,y^-\sim\mathcal{D}}[\log \sigma\big(r_\theta(x, y^+) - r_\theta(x, y^-)\big)] \tag{4}$$

where $\sigma(z) = 1/(1 + e^{-z})$ is the sigmoid function.

Reward-SFT. We introduce Reward-SFT as a control baseline to isolate the effect of the implicit reward from the influence of negative samples. Unlike SFT, which directly maximizes the token-level likelihood, this objective simply maximizes the probability of the chosen response y^+ being classified as "correct" under the implicit reward formulation:

$$\mathcal{L}_{\text{RW-SFT}} = -\mathbb{E}_{x,y^+\sim\mathcal{D}}[\log \sigma\big(r_\theta(x, y^+)\big)] \tag{5}$$

Functionally, this treats the optimization as a binary classification task where the model learns to classify y^+ as "high reward" relative to the reference distribution. As illustrated in Fig. 2, we observe a significant divergence in generalization. SFT+ suffers from immediate catastrophic forgetting, with IFEval performance decaying monotonically as training progresses. DPO remains stable and even improves slightly. Crucially, similar to SFT+, Reward-SFT does not see negative data, yet it matches the OOD stability of DPO. This suggests that the reference-relative objective contributes to the observed retention, motivating the gradient analysis below.

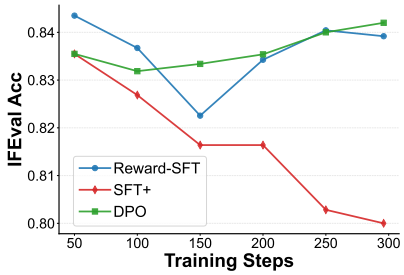


Figure 2: **IFEval acc results (avg@5).** SFT+ forgets instruction following ability, while reward-based methods do not.

3.2 Gradient Analysis: The Mechanics of Tethering

To examine a possible mechanism behind this retention difference, we compare the two objectives at the same parameters and on the same positive sample. Their per-sample gradients differ only by *the dynamic scaling coefficient of the gradient*.

The gradient for SFT with respect to the model parameters θ is:

$$\nabla_\theta \mathcal{L}_{\text{SFT}} = -\nabla_\theta \log \pi_\theta(y^+|x) \tag{6}$$

Meanwhile, the gradient for Reward-SFT in Eq. (5) includes a dynamic re-weighting coefficient determined by r_θ from Eq. (3):

$$\nabla_\theta \mathcal{L}_{\text{RW-SFT}} = -\frac{\partial \log \sigma(r)}{\partial r} \nabla_\theta r_\theta(x, y^+) = -\big(1 - \sigma(r_\theta(x, y^+))\big) \cdot \beta \nabla_\theta \log \pi_\theta(y^+|x) \tag{7}$$

Comparing Equations (6) and (7) reveals a fundamental difference in gradient weighting. In standard SFT, the gradient is scaled by a constant factor of 1. Consequently, SFT uses the same external coefficient for all samples, regardless of their improvement relative to the reference. Even if the model has assigned high probability to the training sample (e.g., $p(y^+|x) \approx 0.99$), the SFT objective continues to maximize its likelihood without an additional reference-dependent attenuation factor. Reward-SFT instead reduces this coefficient as the implicit reward increases, providing adaptive, sample-wise regularization.

In contrast, Reward-SFT introduces an instance-dependent dynamic scaling coefficient $\lambda(x, y^+) = 1 - \sigma(r_\theta(x, y^+))$ (for simplicity, we assume $\beta = 0.1$ in our analysis). We term λ the *Elastic Tether*, as it modulates the "tension" of gradient updates. Functioning as an adaptive regularizer based on the implicit reward, it creates two distinct training modes. **Acquisition (slack tether):** when π_θ is close to π_{ref}, the reward $r_\theta \approx 0$ and $\lambda \approx 0.5$, allowing rapid adaptation to the preferred distribution. **Saturation (taut tether):** as the model aligns with the target and r_θ grows, $\lambda = 1 - \sigma(r_\theta) \to 0$, attenuating further updates on that sample as its likelihood increases relative to the reference.

Consider a sample where the model has achieved high confidence relative to the reference, yielding a reward of $r_\theta = 10$. The sigmoid function saturates, and the gradient scaling coefficient λ becomes:

$$\lambda = 1 - \sigma(10) = 1 - \frac{1}{1 + e^{-10}} \approx 4.5 \times 10^{-5}$$

With $\beta = 0.1$, the full gradient coefficient is $\beta\lambda \approx 4.5 \times 10^{-6}$, compared with 1 for SFT. At the same parameters, this sharply reduces the gradient contribution from the already favored sample.

The Elastic Tether enables a self-regulating process that progressively suppresses updates as a response becomes more likely relative to the reference, acting as a soft, sample-wise early stopping mechanism. This adaptive attenuation provides implicit regularization; the experiments in Section 3.1 demonstrate improved capability retention under Reward-SFT on the same training data.

These analytical insights are supported by the training loss trajectories in Fig. 3 and the reward evolution in Fig. 4. As shown in Fig. 3, the Reward-SFT loss rapidly approaches zero, consistent with Eq. (5) as $r_\theta(x, y^+)$ increases. Furthermore, the reward trajectories align with our gradient analysis: the chosen reward under Reward-SFT plateaus as its gradient scaling coefficient decreases, whereas the SFT+ counterpart continues to rise over the observed training period. Together with the IFEval results in Fig. 2 and the additional SFT+ controls in Appendix D.2, these observations support the role of reward-induced implicit regularization in mitigating forgetting on the same training data. These results show that the optimization objective also matters for retention, complementing studies emphasizing the role of on-policy data (Chen et al., 2025; Shenfeld et al., 2025).

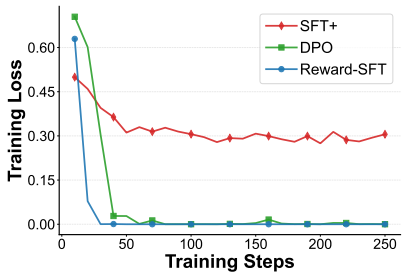


Figure 3: **Training loss curve.** Reward-SFT and DPO converge rapidly; SFT+ remains high under its negative log-likelihood objective.

4 The Surgical Optimization Objective

While preserving prior knowledge is essential, it is insufficient; the model must also achieve substantial in-domain reasoning improvements. This section uncovers the specific failure modes hindering this goal: the "pull-up" effect inherent in positive-only training (e.g., SFT, Reward-SFT) and the inadequacy of relative ranking (DPO) for reasoning with verifiable truth. To bridge this gap, we introduce a reward-based binary cross-entropy objective.

4.1 The "Pull-Up" Effect

While Reward-SFT preserves instruction following in these experiments, it shares a key vulnerability with SFT when trained on our surgical rectification data. As shown in the right panel of Fig. 4, although we train only on chosen data y^+, the likelihood of rejected responses y^- for both SFT+ and Reward-SFT *inadvertently rises relative to the reference model*.

This effect is termed "pull-up" by Ren & Sutherland (2025): under *positive-only* supervision, the model indiscriminately raises the probability mass around the target, reinforcing similar but unwanted responses. In our rectification pipeline, the generated y^+ and y^- retain substantial token overlap. Formally, let $y^\pm = p \oplus s^\pm$, where p is the shared prefix and $s^\pm$ are the diverging suffixes. The gradient of the positive sample decomposes as $\nabla_\theta \log \pi_\theta(y^+|x) = \nabla_\theta \log \pi_\theta(p|x) + \nabla_\theta \log \pi_\theta(s^+|p, x)$. Positive-only training therefore supplies an upward gradient component for the shared-prefix log-probability, but no explicit penalty for the erroneous suffix s^-. This provides a mechanism for the observed pull-up effect and motivates negative supervision at the point of divergence.

4.2 The Insufficiency of Relative Ranking

Given the need for negative supervision, DPO appears to be a natural candidate. DPO models preference probabilities using the Bradley-Terry model (Bradley & Terry, 1952), minimizing the

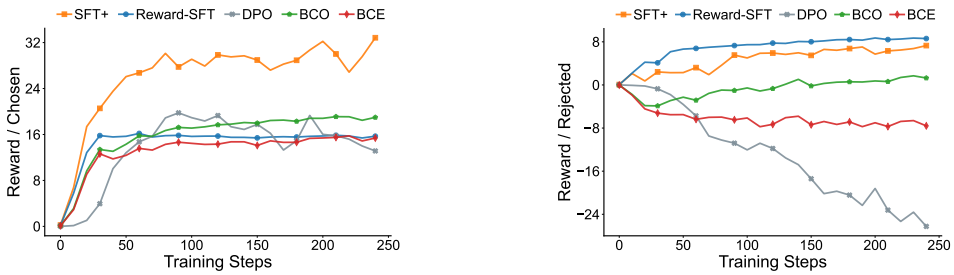


Figure 4: **Evolution of implicit rewards during training.** Left: reward scores for chosen responses; Right: reward scores for rejected responses. The reward measures how much the model's preference for the chosen response has grown relative to the initial state.

negative log-likelihood of the margin $M = r_\theta(x, y^+) - r_\theta(x, y^-)$:

$$P(y^+ \succ y^-) = \sigma(r_\theta(x, y^+) - r_\theta(x, y^-)) \tag{8}$$

However, relative ranking alone can be insufficient for reasoning, where answers are objectively correct or incorrect, unlike subjective tasks (e.g., creative writing) where being "better" suffices. Reasoning requires maximizing the likelihood of the correct response y^+ while suppressing the incorrect y^-. Since gradient descent naturally follows the path of least resistance, the DPO loss can be minimized by decreasing the rejected reward $r(x, y^-)$ while leaving the chosen reward $r(x, y^+)$ stagnant—or even allowing it to drop, provided $r(x, y^-)$ drops faster. This has been observed by Rafailov et al. (2024); Pal et al. (2024); Ren & Sutherland (2025) and corroborated by our results in Fig. 4: in later stages of DPO training, the reward for chosen responses fails to show sustained growth, while the reward for rejected ones decreases drastically. This suggests that, in our experiments, DPO tends to optimize the margin by suppressing y^- rather than reinforcing correct logic, limiting its in-domain reasoning gains.

4.3 Reward-based Binary Cross Entropy Optimization

To address these limitations, we reframe reasoning optimization as a binary classification task. Unlike DPO's relative ranking between pairs, our approach decouples the supervision into independent positive and negative terms. We introduce two variants:

SPoT-BCE. By decoupling the DPO objective in Eq. (4), we drive the model to independently maximize confidence in y^+ and minimize confidence for y^-. This results in the standard Binary Cross Entropy (BCE) loss:

$$\mathcal{L}_{\text{SPoT-BCE}} = -\mathbb{E}_{\mathcal{D}}[\log \sigma(r_\theta(x, y^+)) + \log \sigma(-r_\theta(x, y^-))] \tag{9}$$

The implicit reward $r_\theta(x, y)$ here serves as the classification logit. This formulation provides a dense supervision signal while effectively retaining pre-trained knowledge.

SPoT-BCO. While Binary Classifier Optimization (BCO) (Jung et al., 2025) was originally designed for unpaired data (e.g., thumbs-up/down only), we identify it as well suited for our paired reasoning data. BCO introduces a reward-shift δ to the BCE loss:

$$\mathcal{L}_{\text{SPoT-BCO}} = -\mathbb{E}_{x,y^+,y^-\sim\mathcal{D}}\left[\log \sigma\left(r_\theta(x, y^+) - \delta\right) + \log \sigma\left(-\left(r_\theta(x, y^-) - \delta\right)\right)\right] \tag{10}$$

where $\delta = \frac{1}{2} \cdot \mathbb{E}_{x,y^+,y^-\sim\mathcal{D}}[r_\theta(x, y^+) + r_\theta(x, y^-)]$. In practice, δ is computed as an exponential moving average of the batch statistics, and a stop-gradient is applied so that δ does not contribute to the backward pass (i.e., δ is treated as a constant when computing $\nabla_\theta \mathcal{L}_{\text{SPoT-BCO}}$).

Our contribution lies not in the BCE/BCO loss forms themselves, but in identifying their decoupled structure as well suited to learning from corrected and erroneous reasoning responses, and in analyzing how δ affects gradient weighting.

4.4 Role of the Reward Shift δ

The difference between SPoT-BCE and SPoT-BCO lies in the reward shift δ. While Jung et al. (2025) frame δ as providing a tighter upper bound on DPO, we offer a different viewpoint and analyze its specific role in reasoning alignment.

Table 1: **Qwen3-8B: main results and objective comparisons.** SFT, RFT, and SFT+ use rejection-sampled Gemini 2.5 Pro responses, rejection-sampled self-generated responses, and rectified positive responses, respectively. Despite improving reasoning, SFT+ degrades instruction following. SPoT-BCO achieves the highest overall and in-domain averages, while SPoT-BCE achieves the highest IFEval accuracy. We report avg@16 for AIME24/AIME25/AMC23 and avg@5 otherwise, in percent. The overall Avg equally weights all nine benchmarks. Arrows show changes from Base in percentage points.

Method	In-domain Reasoning							OOD Reasoning			General	Avg
	AIME24	AIME25	AMC23	Math500	Minerva	Olympia	**Avg**	GPQA-D	Connect4	**Avg**	IFEval	
Base	22.1	19.4	66.6	82.8	37.9	52.3	46.9	48.9	10.9	29.9	83.0	47.1
SFT	15.2	13.3	55.9	78.2	38.6	44.3	40.9 ↓6.0	39.0	12.0	25.5 ↓4.4	79.6 ↓3.4	41.8 ↓5.3
RFT	27.3	20.0	59.1	83.0	40.4	53.8	47.3 ↑0.4	49.1	3.1	26.1 ↓3.8	81.5 ↓1.5	46.4 ↓0.7
SFT+	29.4	24.8	67.3	84.8	**42.5**	54.5	50.6 ↑3.7	47.3	14.1	30.7 ↑0.8	80.0 ↓3.0	49.4 ↑2.3
DFT	24.0	23.3	70.5	82.4	40.4	52.4	48.8 ↑1.9	45.6	3.7	24.7 ↓5.2	79.5 ↓3.5	46.9 ↓0.2
DPO	22.1	19.4	62.5	85.2	40.8	51.6	46.9	48.9	31.8	40.4 ↑10.5	84.7 ↑1.7	49.7 ↑2.6
Smaug	22.1	18.8	67.5	84.0	41.9	53.3	47.9 ↑1.0	46.7	34.8	40.8 ↑10.9	84.8 ↑1.8	50.4 ↑3.3
KTO	22.1	18.8	62.5	81.6	39.3	49.6	45.7 ↓1.2	46.9	23.0	35.0 ↑5.1	83.4 ↑0.4	47.5 ↑0.4
Reward-SFT	20.0	16.7	60.5	83.0	39.0	52.3	45.3 ↓1.6	49.2	35.2	**42.2** ↑12.3	83.7 ↑0.7	48.8 ↑1.7
SPoT-BCE	**33.3**	25.2	67.0	**87.6**	40.1	55.4	51.4 ↑4.5	**49.5**	31.1	40.3 ↑10.4	**85.8** ↑2.8	52.8 ↑5.7
SPoT-BCO	27.9	**27.3**	**71.6**	87.4	39.7	**58.5**	**52.1** ↑5.2	47.8	**36.0**	41.9 ↑12.0	84.8 ↑1.8	**53.4** ↑6.3

The implicit reward $r_\theta(x, y)$ defined in Eq. (3) is a simplified version of the theoretical optimal reward $r'(x, y)$. The relationship derived from the KL-constrained optimal policy π^* is given by:

$$r'(x, y) = \beta \log \frac{\pi^*(y|x)}{\pi_{\text{ref}}(y|x)} + \beta \log Z(x) \tag{11}$$

where $Z(x)$ is the partition function. While DPO eliminates $\beta \log Z(x)$ via paired subtraction, BCO admits no analogous cancellation. In Eq. (10), δ sets an adaptive decision threshold at the running mean reward, so the sigmoid weights depend on each response's reward relative to this threshold.

Mitigating Saturation for In-Domain Reasoning. As training progresses, the chosen-response reward $r_\theta(x, y^+)$ may increase (see Fig. 4). In SPoT-BCE ($\delta = 0$), elevated chosen rewards push the positive-term sigmoid into saturation ($\sigma(\cdot) \to 1$), attenuating its gradient. This saturation may slow further reinforcement of chosen responses. By adapting δ via the exponential moving average, SPoT-BCO dynamically adjusts the threshold to enable further updates of the positive term $\log \sigma\left(r_\theta(x, y^+) - \delta\right)$, yielding higher rewards for chosen responses than SPoT-BCE (see Fig. 4) and, consequently, better in-domain reasoning.

The Trade-off with OOD Generalization. Meanwhile, the fixed threshold in SPoT-BCE progressively attenuates the positive-term gradient as $r_\theta(x, y^+)$ increases. An upward adaptive shift of the threshold in SPoT-BCO can sustain these positive-sample updates, consistent with the contrast in chosen-reward trajectories. Empirically, SPoT-BCE achieves higher instruction-following accuracy, whereas SPoT-BCO achieves higher average in-domain reasoning accuracy (Tab. 1), demonstrating a reasoning–retention trade-off in our experiments.

4.5 Surgical Post-Training

Surgical Post-Training takes its name from the synergistic design of our data pipeline and optimization objective. (1) Our data pipeline yields pairs with substantial token overlap and localized corrections to reasoning errors. (2) For tokens in the shared prefix of y^+ and y^-, the gradient contributions from the positive and negative loss terms point in opposite directions, with reward-dependent weights. This can reduce the net update on the shared prefix while retaining separate learning signals for the *divergent* suffixes, acting as a form of *"surgery"* on the model—correcting reasoning errors while limiting changes to the original distribution.

5 Experiments

We present the main results of SPoT, ablate its optimization objective (Sec. 5.2) and data pipeline (Sec. 5.3), and evaluate its use for RL initialization (Sec. 5.4) and post-training beyond mathematics (Sec. 5.5). The mathematical training experiments use the English subset of DAPO-Math-17k (Yu et al., 2025). Full experimental settings are provided in Appendix B.

Table 2: **Results on Llama-3.1-8B-Instruct.** Baseline definitions and evaluation conventions follow Table 1. SPoT-BCO achieves the highest overall, in-domain, and OOD averages while maintaining near-base instruction-following accuracy. Arrows show changes from Base in percentage points.

Method	In-domain Reasoning							OOD Reasoning			General	Avg
	AIME24	AIME25	AMC23	Math500	Minerva	Olympia	**Avg**	GPQA-D	Connect4	**Avg**	IFEval	
Base	3.3	**2.7**	19.5	47.0	22.4	**16.9**	18.6	28.5	5.0	16.8	**73.6**	24.3
SFT	3.3	2.1	21.6	46.8	19.5	15.1	18.1 ↓0.5	29.6	1.7	15.7 ↓1.1	62.1 ↓11.5	22.4 ↓1.9
RFT	3.3	0.0	24.1	46.0	19.1	15.3	18.0 ↓0.6	**32.3**	2.0	17.2 ↑0.4	71.2 ↓2.4	23.7 ↓0.6
SFT+	3.3	0.6	25.9	**48.6**	25.7	15.3	19.9 ↑1.3	31.9	1.5	16.7 ↓0.1	68.6 ↓5.0	24.6 ↑0.3
SPoT-BCO	**4.0**	2.1	**27.0**	**48.6**	**26.1**	16.3	**20.7** ↑2.1	30.0	**7.0**	**18.5** ↑1.7	73.2 ↓0.4	**26.0** ↑1.7

5.1 Results

We compare SPoT with the unmodified model (Base) and three supervised fine-tuning baselines: (1) **SFT:** supervised fine-tuning on rejection-sampled responses from Gemini 2.5 Pro. (2) **RFT:** rejection sampling fine-tuning using self-generated responses. (3) **SFT+:** supervised fine-tuning on the rectified positive samples y^+. On Qwen3-8B, we also compare SPoT-BCE and SPoT-BCO against DFT (Wu et al., 2026), DPO, Smaug (Pal et al., 2024), KTO (Ethayarajh et al., 2024), and Reward-SFT (Table 1); Sec. 5.2 analyzes these objectives. Table 2 reports results on Llama-3.1-8B-Instruct.

Catastrophic forgetting under SFT. SFT compromises general capabilities and OOD reasoning. This degradation is even more pronounced on Llama-3.1-8B-Instruct, where IFEval accuracy drops by 11.5 points. Furthermore, the drastic distributional shift between Gemini 2.5 Pro and the policy models erodes even in-domain reasoning (e.g., Qwen3-8B drops from 46.9% to 40.9%). These results demonstrate that direct SFT on off-policy data induces substantial forgetting of prior capabilities.

Proximity alone is insufficient. Despite using "surgical" data proximal to the policy model, SFT+ improves in-domain reasoning yet fails to prevent forgetting on general tasks. Even with more "on-policy" data, RFT yields negligible in-domain improvements but incurs notable forgetting. These results indicate that data proximity alone is insufficient to prevent forgetting under the SFT objective.

SPoT: superior reasoning with retained knowledge. SPoT-BCO leads the evaluated methods in overall and in-domain averages on both backbones, while improving OOD reasoning over each base model. It improves instruction following on Qwen3-8B and maintains near-base instruction-following accuracy on Llama-3.1-8B-Instruct. This demonstrates the effectiveness of both our surgical data rectification pipeline and our optimization objective. Additional general-capability results on TruthfulQA and MMLU-Pro are reported in Appendix D.4.

5.2 Comparison on Loss Objectives

In Table 1, SPoT-BCO leads overall (53.4%) and in-domain (52.1%) averages; SPoT-BCE leads IFEval (85.8%) and GPQA-D (49.5%). BCO's adaptive threshold δ can sustain positive-sample updates as implicit rewards increase.

Smaug improves in-domain reasoning over DPO (47.9% versus 46.9%), but remains below SPoT-BCO overall (50.4% versus 53.4%). KTO retains IFEval accuracy (83.4%) yet lowers the in-domain average to 45.7%, yielding 47.5% overall. These results favor SPoT's binary cross-entropy objectives over pairwise ranking and KTO's alternative binary-feedback objective in this setting. Reward-SFT achieves the highest OOD average (42.2%), but its in-domain average declines to 45.3%. DFT improves in-domain reasoning over the base model while reducing OOD reasoning and instruction following. We further compare against the DPO+SFT recipe (Wang et al., 2024) in Appendix D.3.

5.3 The Effect of the Surgical Rectification Pipeline

We validate the impact of *data proximity* by ablating two key components: the data source and the proximity constraint γ in Sec. 2. The results are summarized in Tab. 3. First, training on rectified data outperforms direct data in overall average (+5.2 percentage points). Although SPoT-BCO already limits forgetting on instruction following (only a 0.4-point IFEval drop), the substantial reasoning gain from rectified data confirms that proximal data is inherently more effective. Second, scaling the dataset size from 2k to 4k improves the overall average. Finally, when controlling for the final training set size, the subset filtered with $\gamma = 0.6$ achieves the highest average score (53.4%), surpassing the

Table 3: **Ablation study on different data.** "Rectified data" refers to the dataset curated using our surgical rectification pipeline, while "direct data" uses the rejection-sampled direct answers from Gemini 2.5 Pro. γ defines the threshold of change ratio in Equation (1).

Model	In-domain Reasoning							OOD Reasoning			General	Avg
	AIME24	AIME25	AMC23	Math500	Minerva	Olympia	**Avg**	GPQA-D	Connect4	**Avg**	IFEval	
2k direct data	20.0	14.8	63.0	81.6	37.5	47.6	44.1 ↓2.8	42.6	17.2	29.9	82.6 ↓0.4	45.2 ↓1.9
2k rectified data, $\gamma = 0.6$	26.0	19.4	67.0	**87.8**	41.2	52.4	49.0 ↑2.1	**49.8**	26.0	37.9 ↑8.0	84.1 ↑1.1	50.4 ↑3.3
4k rectified data, $\gamma = 1$	27.3	22.7	64.1	87.6	**41.5**	58.2	50.2 ↑3.3	49.2	33.3	41.3 ↑11.4	83.6 ↑0.6	51.9 ↑4.8
4k rectified data, $\gamma = 0.6$	**27.9**	**27.3**	**71.6**	87.4	39.7	**58.5**	**52.1** ↑5.2	47.8	**36.0**	**41.9** ↑12.0	**84.8** ↑1.8	**53.4** ↑6.3

Table 4: **Comparison with GRPO.** SPoT + GRPO: GRPO initialized from SPoT checkpoint.

Method	AIME24	AIME25	AMC23	Connect4	IFEval	**Avg**
SPoT	27.9 ↑5.8	27.3 ↑7.9	71.6 ↑5.0	36.0 ↑25.1	84.8 ↑1.8	49.5 ↑9.1
GRPO	36.3 ↑14.2	28.8 ↑9.4	77.5 ↑10.9	7.7 ↓3.2	81.2 ↓1.8	46.3 ↑5.9
SPoT + GRPO	41.3 ↑19.2	36.7 ↑17.3	85.9 ↑19.3	29.4 ↑18.5	82.1 ↓0.9	55.1 ↑14.7

unfiltered configuration ($\gamma = 1$). This further confirms that enforcing data proximity is critical for maximizing performance.

5.4 Superior Initialization for Reinforcement Learning

To understand how SPoT affects subsequent RL, we evaluate GRPO training initialized from the SPoT checkpoint (Tab. 4). We observe that GRPO yields more in-domain improvements compared to SPoT, but induces some forgetting relative to the base model, evidenced by performance degradation on Connect4 (our contamination-free OOD benchmark) and a 1.8-percentage-point decrease on IFEval. More importantly, SPoT provides a superior initialization for subsequent RL, significantly elevating the performance ceiling. SPoT + GRPO outperforms GRPO alone across all benchmarks (+7.1 percentage points in in-domain math and +21.7 percentage points on Connect4), alongside better general instruction-following capabilities. The results demonstrate that SPoT's knowledge injection establishes a stronger foundation for RL exploration.

5.5 Post-Training Beyond Mathematics

To evaluate SPoT beyond mathematical post-training, we train Qwen3-8B on 1,000 Connect4 examples rectified by GPT-5.5. We deliberately omit final-answer filtering to test tolerance to Oracle mistakes. SPoT raises Connect4 accuracy from 10.9% to 46.7%, compared with 43.9% for SFT, while retaining higher IFEval accuracy (83.3% versus 80.4%). It also improves AIME24, AIME25, and Math500 over the base model. See Appendix F.2 for details.

6 Conclusion and Discussion

We present SPoT, a post-training framework that combines a *surgical rectification pipeline* with a *binary optimization objective* to efficiently inject reasoning knowledge while mitigating forgetting in our evaluations. Our gradient analysis establishes sample-wise attenuation under the reference-relative reward formulation, while our experiments demonstrate improved capability retention and advantages of binary supervision over DPO in the evaluated setting. With only 4k rectified pairs and minutes of training, SPoT improves average in-domain and OOD reasoning accuracy on Qwen3-8B, and substantially raises the performance ceiling of subsequent GRPO, positioning SPoT as a lightweight bridge between SFT and RL.

Limitations and future work. Due to resource constraints, our experiments focus on smaller models (up to 8B) and text-based mathematical and game reasoning (additional Qwen3-1.7B results in Appendix D.5). Performance also depends on the Oracle's capability; we report a sensitivity analysis in Appendix E.1, leaving a larger-scale study across diverse Oracles for future work. Future work will examine whether SPoT's reasoning and retention benefits extend to larger models and to multimodal reasoning, code generation, and agentic tasks.

AI Use Statement

The authors used AI tools to improve the writing and edit the manuscript. Gemini 2.5 Pro and GPT-5.5 were used to rectify student-generated reasoning traces for the mathematical and Connect4 training experiments, respectively. The authors take responsibility for the final text, analyses, and experimental claims.

Ethics Statement

This work aims to improve reasoning while reducing catastrophic forgetting. Our experiments use existing benchmarks and programmatically generated Connect4 states. Oracle-generated training data may contain errors or inherit model biases. Retaining instruction-following accuracy does not guarantee safe behavior; downstream applications require separate evaluation of safety and fairness.

Reproducibility Statement

We describe the rectification pipeline in Sec. 2 and the optimization objectives in Sec. 4.3. The appendices provide Oracle prompts (Appendix C.1), training and evaluation settings and hyperparameter search spaces (Appendix B), repeated-run results (Appendix D.1), and Connect4 benchmark construction and post-training (Appendices F.1 and F.2). We will release the curated training data, curation and training code, model checkpoints, and evaluation scripts.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. GPT-4 technical report. *arXiv*, 2023.

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 *ICLR*, 2024.

Anthropic. The Claude 3 model family: Opus, Sonnet, Haiku. *Anthropic model card*, 2024.

Mohammad Gheshlaghi Azar, Zhaohan Daniel Guo, Bilal Piot, Remi Munos, Mark Rowland, Michal Valko, and Daniele Calandriello. A general theoretical paradigm to understand learning from human preferences. In *AISTATS*, 2024.

Ralph Allan Bradley and Milton E Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 1952.

Howard Chen, Noam Razin, Karthik Narasimhan, and Danqi Chen. Retaining by doing: The role of on-policy data in mitigating forgetting. *arXiv*, 2025.

Zixiang Chen, Yihe Deng, Huizhuo Yuan, Kaixuan Ji, and Quanquan Gu. Self-play fine-tuning converts weak language models to strong language models. In *ICML*, 2024.

Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V Le, Sergey Levine, and Yi Ma. SFT memorizes, RL generalizes: A comparative study of foundation model post-training. In *ICML*, 2025.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv*, 2025.

Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. Model alignment as prospect theoretic optimization. In *ICML*, 2024.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. MiniLLM: Knowledge distillation of large language models. In *ICLR*, 2024.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, et al. OlympiadBench: A challenging benchmark for promoting AGI with olympiad-level bilingual multimodal scientific problems. In *ACL*, 2024.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *NeurIPS Datasets and Benchmarks Track*, 2021.

Andreas Hochlehnert, Hardik Bhatnagar, Vishaal Udandarao, Samuel Albanie, Ameya Prabhu, and Matthias Bethge. A sober look at progress in language model reasoning: Pitfalls and paths to reproducibility. In *COLM*, 2025.

Jiwoo Hong, Noah Lee, and James Thorne. ORPO: Monolithic preference optimization without reference model. In *EMNLP*, 2024.

Maggie Huan, Yuetai Li, Tuney Zheng, Xiaoyu Xu, Seungone Kim, Minxin Du, Radha Poovendran, Graham Neubig, and Xiang Yue. Does math reasoning improve general LLM capabilities? Understanding transferability of LLM reasoning. *arXiv*, 2025.

Seungjae Jung, Gunsoo Han, Daniel Wontae Nam, and Kyoung-Woon On. Binary classifier optimization for large language model alignment. In *ACL*, 2025.

Yoon Kim and Alexander M. Rush. Sequence-level knowledge distillation. In *EMNLP*, 2016.

Xin Lai, Zhuotao Tian, Yukang Chen, Senqiao Yang, Xiangru Peng, and Jiaya Jia. Step-DPO: Step-wise preference optimization for long-chain reasoning of LLMs. *arXiv*, 2024.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. Solving quantitative reasoning problems with language models. In *NeurIPS*, 2022.

Yaxuan Li, Yuxin Zuo, Bingxiang He, Jinqian Zhang, Chaojun Xiao, Cheng Qian, Tianyu Yu, Huanang Gao, Wenkai Yang, Zhiyuan Liu, and Ning Ding. Rethinking on-policy distillation of large language models: Phenomenology, mechanism, and recipe. *arXiv*, 2026.

Wenye Lin, Jonathan Roberts, Yunhan Yang, Samuel Albanie, Zongqing Lu, and Kai Han. GAME-BoT: Transparent assessment of LLM reasoning in games. In *ACL*, 2025.

Kevin Lu and Thinking Machines Lab. On-policy distillation. *Thinking Machines Lab: Connectionism*, 2025.

Yu Meng, Mengzhou Xia, and Danqi Chen. SimPO: Simple preference optimization with a reference-free reward. In *NeurIPS*, 2024.

Sagnik Mukherjee, Lifan Yuan, Dilek Hakkani-Tur, and Hao Peng. Reinforcement learning finetunes small subnetworks in large language models. *arXiv*, 2025.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. In *NeurIPS*, 2022.

Arka Pal, Deep Karkhanis, Samuel Dooley, Manley Roberts, Siddartha Naidu, and Colin White. Smaug: Fixing failure modes of preference optimisation with DPO-Positive. *arXiv*, 2024.

Richard Yuanzhe Pang, Weizhe Yuan, He He, Kyunghyun Cho, Sainbayar Sukhbaatar, and Jason Weston. Iterative reasoning preference optimization. In *NeurIPS*, 2024.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *NeurIPS*, 2023.

Rafael Rafailov, Joey Hejna, Ryan Park, and Chelsea Finn. From r to Q^*: Your language model is secretly a Q-function. In *COLM*, 2024.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. GPQA: A graduate-level Google-proof Q&A benchmark. In *COLM*, 2024.

Yi Ren and Danica J Sutherland. Learning dynamics of LLM finetuning. In *ICLR*, 2025.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv*, 2024.

Idan Shenfeld, Jyothish Pari, and Pulkit Agrawal. RL's razor: Why online reinforcement learning forgets less. *arXiv*, 2025.

Mingyang Song and Mao Zheng. A survey of on-policy distillation for large language models. *arXiv*, 2026.

Robert A Wagner and Michael J Fischer. The string-to-string correction problem. *JACM*, 1974.

Weiyun Wang, Zhe Chen, Wenhai Wang, Yue Cao, Yangzhou Liu, Zhangwei Gao, Jinguo Zhu, Xizhou Zhu, Lewei Lu, Yu Qiao, et al. Enhancing the reasoning ability of multimodal large language models via mixed preference optimization. *arXiv*, 2024.

Yongliang Wu, Yizhou Zhou, Ziheng Zhou, Yingzhe Peng, Xinyu Ye, Xinting Hu, Wenbo Zhu, Lu Qi, Ming-Hsuan Yang, and Xu Yang. On the generalization of SFT: A reinforcement learning perspective with reward rectification. In *ICLR*, 2026.

Bangjun Xiao, Bingquan Xia, Bo Yang, Bofei Gao, Bowen Shen, Chen Zhang, Chenhong He, Chiheng Lou, Fuli Luo, Gang Wang, et al. MiMo-V2-Flash technical report. *arXiv*, 2026.

Hongling Xu, Qi Zhu, Heyuan Deng, Jinpeng Li, Lu Hou, Yasheng Wang, Lifeng Shang, Ruifeng Xu, and Fei Mi. KDRL: Post-training reasoning LLMs via unified knowledge distillation and reinforcement learning. *arXiv*, 2025.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv*, 2025.

Tianzhu Ye, Li Dong, Zewen Chi, Xun Wu, Shaohan Huang, and Furu Wei. Black-box on-policy distillation of large language models. *arXiv*, 2025.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. DAPO: An open-source LLM reinforcement learning system at scale. *arXiv*, 2025.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in LLMs beyond the base model? *arXiv*, 2025.

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. *arXiv*, 2026.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv*, 2023.

A Related Work

Catastrophic Forgetting. Recent work highlights a divergence between SFT and RL: while SFT is prone to overfitting and catastrophic forgetting, RL demonstrates superior generalization (Chu et al., 2025; Huan et al., 2025). To explain this, Mukherjee et al. (2025) observed that RL updates are typically sparse, affecting only specific parameter subnetworks, whereas SFT triggers dense, global updates. Alternatively, Shenfeld et al. (2025) and Chen et al. (2025) argued that on-policy data generation—rather than the optimization objective—is the primary driver of knowledge retention. However, our results in Sec. 3 show that the training objective also affects knowledge retention: reward-based objectives preserve prior knowledge better than SFT on the same proximal data. Our gradient analysis further identifies adaptive gradient attenuation as an implicit regularization mechanism in reward-based training.

DPO for Reasoning. DPO tends to overly penalize rejected responses rather than strengthening the chosen ones (Ren & Sutherland, 2025; Azar et al., 2024; Pal et al., 2024), making it sub-optimal for reasoning. To mitigate this, several variants have emerged: Azar et al. (2024) introduced IPO to regularize against overfitting; Pal et al. (2024) and Meng et al. (2024) encouraged a larger margin to incentivize learning from positive examples; and others (Hong et al., 2024; Wang et al., 2024) integrated SFT objectives to reinforce preferred responses.

Furthermore, methods such as KTO (Ethayarajh et al., 2024) and BCO (Jung et al., 2025) use unpaired or binary feedback to reduce annotation costs. While BCO was originally designed for data efficiency, we show, both analytically and empirically, that its loss structure is well suited to address the ranking inefficiencies of DPO while mitigating catastrophic forgetting during post-training for reasoning.

Beyond general alignment, other works also leverage DPO to enhance reasoning capabilities. Chen et al. (2024) and Pang et al. (2024) employed iterative self-improvement, distinguishing model generations from human data. Lai et al. (2024) introduced Step-DPO, which includes a data rectification pipeline for precise step-level feedback. SPoT distinguishes itself from Step-DPO in two critical aspects. First, Step-DPO uses student-generated corrections, with humans or GPT-4 assisting error localization and verification. In SPoT, the Oracle directly produces minimally edited corrections, without requiring successful student resampling. Second, whereas Step-DPO retains the standard DPO objective, SPoT uses a reward-based binary cross-entropy objective. This approach more effectively addresses the sub-optimality of DPO for reasoning while mitigating forgetting in the post-training phase.

On-Policy Distillation. SPoT can be viewed as a form of on-policy distillation (OPD), in which the student is trained to recover from its own self-generated mistakes under teacher supervision (Agarwal et al., 2024; Gu et al., 2024). Recent analyses (Li et al., 2026; Song & Zheng, 2026) and large-scale practice (Yang et al., 2025; Lu & Thinking Machines Lab, 2025) confirm that OPD delivers substantial reasoning enhancements with high computational efficiency, validating the on-policy-with-dense-supervision recipe. However, these approaches rely on logit-level supervision from the teacher, which is unavailable for the strongest proprietary models accessible only via inference APIs. Existing black-box workarounds either fall back to off-policy SeqKD (Kim & Rush, 2016) on teacher-generated text, or, more recently, replace the missing logit signal with an adversarially trained on-policy reward model (Ye et al., 2025). Orthogonally, OPSD (Zhao et al., 2026) dispenses with an external teacher altogether by letting a single model play both roles under different conditioning contexts.

SPoT sits in this design space at a deliberately different point. First, it operates strictly under *query-based API access*—requiring neither teacher logits nor a learned discriminator—which makes it directly compatible with frontier proprietary teachers and sidesteps the well-known instability of adversarial training in Ye et al. (2025). Second, in place of the standard divergence-matching loss, SPoT adopts a *reward-based binary cross-entropy* objective explicitly designed to mitigate catastrophic forgetting. Our evaluation focuses on its reasoning–retention trade-off, complementing studies of hyperparameter-sensitive RL/KL objectives (Xu et al., 2025). Unlike Zhao et al. (2026), SPoT admits genuinely stronger external teachers rather than constraining the student to its own conditional distribution.

B EXPERIMENTAL SETTINGS

Models and Datasets. We adopt Qwen3-8B and Llama-3.1-8B-Instruct as the policy models for training. We intentionally choose instruction-tuned models over base models, since these already post-trained models are more susceptible to catastrophic forgetting (Lu & Thinking Machines Lab, 2025). If SPOT retains these models' prior knowledge during subsequent training, it validates the robustness of our approach. For Qwen3-8B, we enforce a non-thinking mode by explicitly inserting an empty thinking block ("`<think>\n\n</think>\n\n`"). We leverage Gemini 2.5 Pro as the Oracle for rectification.

Experiments are conducted using the English subset of DAPO-Math-17k (Yu et al., 2025). Leveraging our pipeline, we generate 4k contrastive pairs for Qwen3-8B. For Llama-3.1-8B-Instruct model, we randomly sample 1.5k pairs, matching the number of correct responses the model could inherently generate, to ensure a fair comparison with Rejection sampling Fine-Tuning (RFT). All methods are compared using the same amount of training data.

Evaluation. We evaluate performance across: *in-domain reasoning*, including AIME24, AIME25, AMC23, Math500 (Hendrycks et al., 2021), Minerva (Lewkowycz et al., 2022), and Olympia (He et al., 2024); *OOD reasoning*, including GPQA-Diamond (GPQA-D) (Rein et al., 2024) and `Connect4`, and *general instruction following* using IFEval (Zhou et al., 2023). To strictly evaluate OOD reasoning without data contamination, we leverage GAMEBoT (Lin et al., 2025) to dynamically construct the `Connect4` dataset (see Appendix F.1). For reproducibility and fairness, we adopt the evaluation prompts from SoberEval (Hochlehnert et al., 2025). We search for the optimal evaluation hyperparameters for the base models and apply them consistently across all compared methods.

Hyperparameters. For data generation and evaluation, we employ a temperature of 0.7 and top-p of 0.8, with a maximum token limit of 32,768. For the main post-training comparisons, we train models for 2 epochs with a context length of 8,192 and $\beta = 0.1$. To ensure a fair comparison, we systematically tune the hyperparameters for the baselines. Regarding learning rates, SPOT utilizes 1×10^{-6} for Qwen3-8B and 2×10^{-7} for Llama-3.1-8B-Instruct. For SFT, RFT, and SFT+, we use learning rates of 5×10^{-6} (Qwen3-8B) and 1×10^{-6} (Llama-3.1-8B-Instruct). The batch size is set to 32 for all the methods.

Hyperparameter Search Space. For SFT, RFT, SFT+, and DFT, we search learning rates in $\{10^{-6}, 2 \times 10^{-6}, 5 \times 10^{-6}, 10^{-5}, 2 \times 10^{-5}\}$ and training durations of $\{1, 2, 3, 5\}$ epochs. For DPO, Smaug, KTO, Reward-SFT, and SPOT, we search learning rates in $\{10^{-7}, 2 \times 10^{-7}, 5 \times 10^{-7}, 10^{-6}, 2 \times 10^{-6}, 5 \times 10^{-6}\}$, the same epoch choices, and $\beta \in \{0.1, 0.2, 0.5\}$.

RL Experiment Setting. For the RL experiment in Sec. 5.4, we adopt GRPO (Shao et al., 2024) implemented in `verl` v0.6.0 with vLLM v0.9.0 as the rollout backend. We use 4k English subset of DAPO-Math-17k for RL training. All runs are conducted on a single node with 8× H800 (80GB) GPUs. For each prompt we draw a group of n=8 rollouts at temperature 1.0. The training batch size is 256 with a mini-batch size of 64, and we train for 10 epochs with a learning rate of 1×10^{-6}. For Qwen3-8B we keep the non-thinking mode by passing `enable_thinking=False` to the chat template, consistent with the SPOT setup. We set the response length limit to 4096 to fit in memory.

C Data Rectification Details

C.1 Rectification Prompt

Rectification Prompt Without Ground Truth

```
Act as a helpful teaching assistant. Your goal is to revise a student model's answer to make it correct,
while maintaining the student model's original writing style, tone, and formatting. The final result
should look as if the student model had solved the problem correctly on its first try.

You should first solve the problem independently and do the following:
1. Identify the correct parts of the student model's answer and keep them.
2. Replace the incorrect parts with correct reasoning.
3. Carefully match the student model's original writing style, including their tone, vocabulary,
formatting and sentence structure.

**IMPORTANT OUTPUT FORMAT:**
1. First output ``=== CORRECTED STARTED ==='' followed by the corrected answer
2. Ends with the corrected answer in the format: 'Therefore, the final answer is: $\\boxed{{ANSWER}}$.'
3. Then output ``=== CORRECTED ENDED ==='' at the end of the corrected trace
4. Do not output meta-phrases like "Here is the corrected version"
```

Rectification Prompt With Ground Truth

```
Act as a helpful teaching assistant. Your goal is to revise a student model's answer to make it correct,
while maintaining the student model's original writing style, tone, and formatting. The final result
should look as if the student model had solved the problem correctly on its first try.

You should compare the student model's answer with the Reference Ground Truth and do the following:
1. Identify the correct parts of the student model's answer and keep them.
2. Replace the incorrect parts with correct reasoning.
3. Carefully match the student model's original writing style, including their tone, vocabulary, and
sentence structure.

**IMPORTANT OUTPUT FORMAT:**
1. First output ``=== CORRECTED STARTED ==='' followed by the corrected answer
2. Ends with the corrected answer in the format: 'Therefore, the final answer is: $\\boxed{{ANSWER}}$.'
3. Then output ``=== CORRECTED ENDED ==='' at the end of the corrected trace
4. Do not output meta-phrases like "Here is the corrected version"
```

C.2 Change Ratio of Rectification

We visualize the distribution of the token-level LCS-based change ratio R_{LCS} defined in Eq. (1). It measures the fraction of tokens in the rectified response y^+ that are not included in its LCS with the original response y^-. The results are shown in Fig. 5.

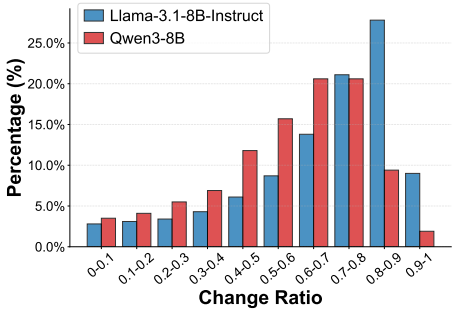


Figure 5: **Distributions of change ratio.** Higher values indicate less token overlap between the original and rectified responses. The distribution shape is determined by the model ability and the data difficulty together.

C.3 RECTIFICATION SAMPLES

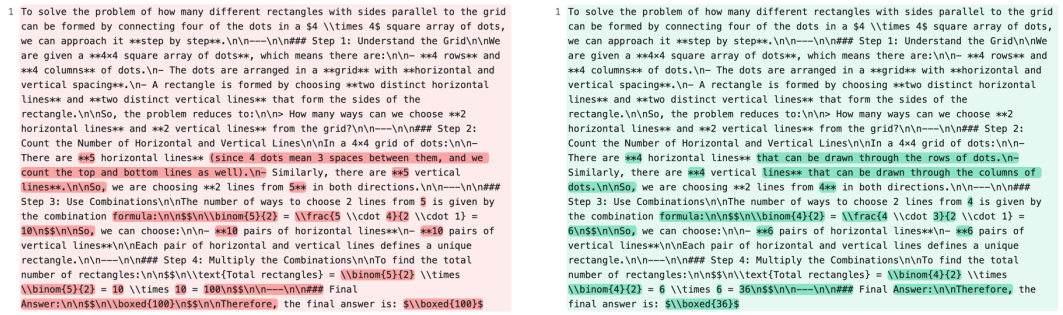


Figure 6: **Random Rectification Example 1.** Left: answer from Qwen3-8B; Right: rectification by Gemini 2.5 Pro.



Figure 7: **Random Rectification Example 2.** Left: answer from Qwen3-8B; Right: rectification by Gemini 2.5 Pro.

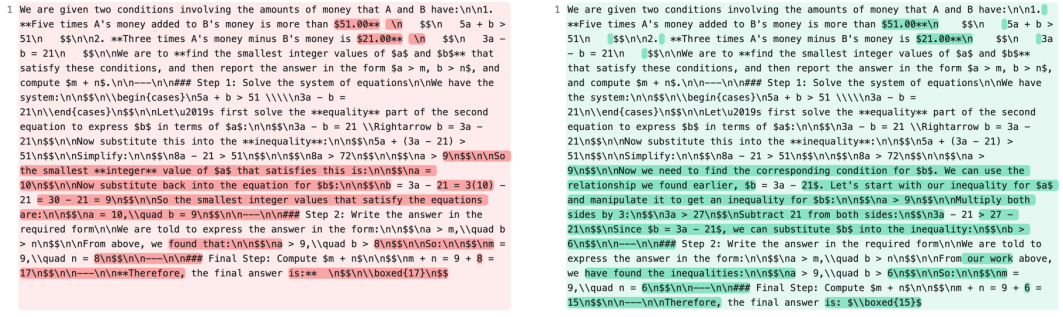


Figure 8: **Random Rectification Example 3.** Left: answer from Qwen3-8B; Right: rectification by Gemini 2.5 Pro.

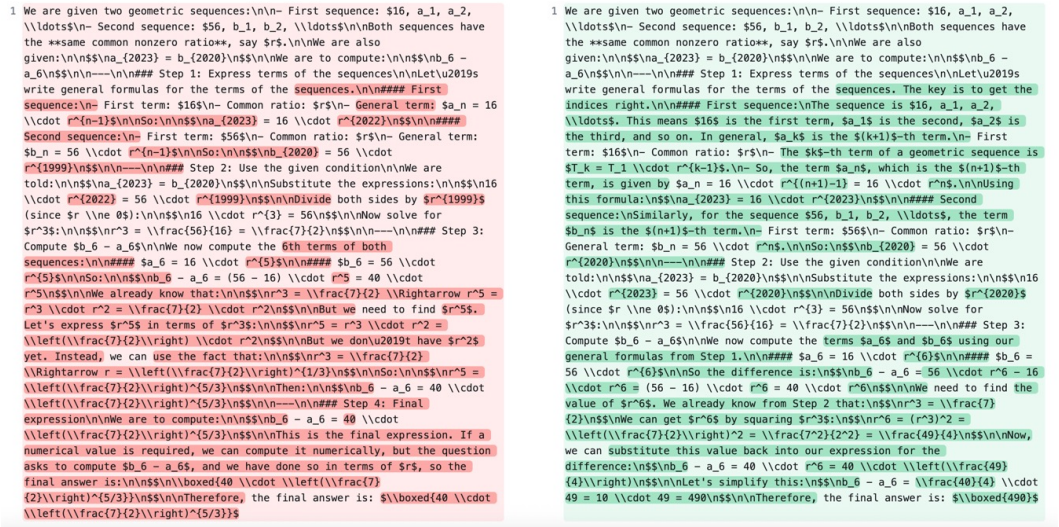


Figure 9: **Random Rectification Example 4.** Left: answer from Qwen3-8B; Right: rectification by Gemini 2.5 Pro.

D Additional Results and Controls

D.1 Detailed Results with Standard Deviations

Table 5 reports detailed results and sampling standard deviations for Qwen3-8B Base, SFT, RFT, SFT+, and SPoT-BCO. For all generative evaluations, the error bars (standard deviations) are calculated across k independent sampling runs. Specifically, we compute the standard deviation over $k = 16$ runs for AIME24, AIME25, and AMC23, and over $k = 5$ runs for the remaining datasets.

For each trained checkpoint, these sampling runs repeat decoding while holding the model weights fixed. We additionally report results from five SPoT runs in Table 6, with the mean and standard deviation computed across the five runs.

Table 5: **Detailed results with standard deviations.** This table reports a subset of the Qwen3-8B results in Table 1. The standard deviations are calculated across independent sampling runs ($k = 16$ for AIME24, AIME25, and AMC23, and $k = 5$ for the others). The average summary columns are omitted here for brevity.

Model	In-domain Reasoning						OOD Reasoning		General
	AIME24	AIME25	AMC23	Math500	Minerva	Olympia	GPQA-D	Connect4	IFEval
Qwen3-8B	$22.1_{\pm 1.7}$	$19.4_{\pm 1.2}$	$66.6_{\pm 1.7}$	$82.8_{\pm 0.4}$	$37.9_{\pm 0.6}$	$52.3_{\pm 0.5}$	$48.9_{\pm 0.7}$	$10.9_{\pm 1.4}$	$83.0_{\pm 0.3}$
Qwen3-8B+SFT	$15.2_{\pm 1.7}$	$13.3_{\pm 1.4}$	$55.9_{\pm 1.7}$	$78.2_{\pm 0.5}$	$38.6_{\pm 0.7}$	$44.3_{\pm 0.6}$	$39.0_{\pm 1.6}$	$12.0_{\pm 1.5}$	$79.6_{\pm 0.4}$
Qwen3-8B+RFT	$27.3_{\pm 1.4}$	$20.0_{\pm 1.6}$	$59.1_{\pm 2.0}$	$83.0_{\pm 0.4}$	$40.4_{\pm 0.6}$	$53.8_{\pm 0.5}$	$49.1_{\pm 0.9}$	$3.1_{\pm 1.1}$	$81.5_{\pm 0.5}$
Qwen3-8B+**SFT+**	$\mathbf{29.4}_{\pm 0.8}$	$24.8_{\pm 0.8}$	$67.3_{\pm 3.0}$	$84.8_{\pm 0.4}$	$\mathbf{42.5}_{\pm 0.5}$	$54.5_{\pm 0.5}$	$47.3_{\pm 1.2}$	$14.1_{\pm 1.3}$	$80.0_{\pm 0.6}$
Qwen3-8B+**SPoT**	$27.9_{\pm 1.1}$	$\mathbf{27.3}_{\pm 1.1}$	$\mathbf{71.6}_{\pm 1.3}$	$\mathbf{87.4}_{\pm 0.5}$	$39.7_{\pm 0.5}$	$\mathbf{58.5}_{\pm 0.7}$	$47.8_{\pm 0.5}$	$\mathbf{36.0}_{\pm 1.8}$	$\mathbf{84.8}_{\pm 0.4}$

Table 6: Five SPoT runs on Qwen3-8B. All entries are accuracy in percent; the final row reports the mean and sample standard deviation across runs.

Run	In-domain	OOD	IFEval
1	52.45	42.25	85.0
2	51.73	41.55	84.3
3	52.45	40.10	85.4
4	51.92	42.25	84.5
5	51.78	40.85	84.8
Mean $\pm$ SD	52.07 ± 0.36	41.40 ± 0.93	84.80 ± 0.43

D.2 Additional SFT Controls

On the same rectified Qwen3-8B data, weaker SFT+ updates recover some IFEval accuracy at the cost of smaller reasoning gains. An explicit-KL SFT+ control shows a similar trade-off. Table 7 reports the results for smaller learning rates, a shorter training run, and explicit KL regularization.

Table 7: SFT+ controls on the same rectified Qwen3-8B data. In-domain and OOD columns follow the benchmark groups in Appendix B; all entries are accuracy in percent.

Setting	In-domain	OOD	IFEval
Base model	46.9	29.9	83.0
SFT+ (5×10^{-6}, 2 epochs)	50.6	30.7	80.0
SFT+ (1×10^{-6}, 2 epochs)	47.6	30.0	81.2
SFT+ (2×10^{-6}, 2 epochs)	48.6	30.2	80.7
SFT+ (5×10^{-6}, 1 epoch)	49.5	30.5	80.5
SFT+ with explicit KL	48.6	29.8	81.0

D.3 Comparison with DPO+SFT

We additionally compare SPoT against the DPO+SFT recipe (Wang et al., 2024), which combines preference optimization with supervised fine-tuning. Under the same setting on Qwen3-8B (Tab. 8), SPoT outperforms DPO+SFT on every benchmark.

Table 8: Comparison with DPO+SFT (Wang et al., 2024) on Qwen3-8B. Both methods use the same 2k rectified training pairs. SPoT dominates across in-domain math, OOD reasoning, and instruction following.

Model	AIME24	AIME25	AMC23	Connect4	IFEval
Base	22.1	19.4	66.6	10.9	83.0
DPO+SFT (Wang et al., 2024)	24.6	19.0	66.6	21.9	81.5
SPoT (Ours)	**26.0**	**19.4**	**67.0**	**26.0**	**84.1**

D.4 General-Capability Evaluation

A central design goal of SPoT is to inject new knowledge without degrading the model's prior abilities. Beyond the benchmarks reported in the main text, we further evaluate on TruthfulQA and a 500-sample subset of MMLU-Pro (Tab. 9). The results suggest that on-policy rectification introduces useful knowledge while retaining the model's general competencies.

Table 9: General-capability evaluations on Qwen3-8B. SPoT preserves truthfulness and improves on MMLU-Pro.

Model	TruthfulQA	MMLU-Pro (500 subset)
Qwen3-8B Base	68.4	51.2
Qwen3-8B + SPoT	67.9	55.2

D.5 Experiment on Qwen3-1.7B

We additionally apply SPoT to Qwen3-1.7B (Tab. 10). SPoT yields consistent gains across in-domain math and IFEval, indicating that the rectification-based recipe transfers to smaller capacities. `Connect4` is too difficult for both Base and SPoT at this scale, indicating that the model lacks the reasoning capacity for the task rather than reflecting a failure of the training recipe.

Table 10: SPoT on Qwen3-1.7B. The recipe transfers to smaller models.

Model	AIME24	AIME25	AMC23	IFEval	Connect4
Base	12.7	9.4	40.9	67.7	0.0
SPoT	14.6	13.3	45.9	68.3	0.0

E Robustness and Ablations

E.1 Sensitivity to Oracle Capability

We replace the default Oracle (Gemini 2.5 Pro) with the weaker open-source `Qwen3-30B-A3B-2507`; SPoT remains effective (Tab. 11).

Table 11: SPoT with a weaker open-source Oracle (`Qwen3-30B-A3B-2507`) on Qwen3-8B. Even without a frontier API, the framework improves mathematical and game reasoning.

Model	AIME24	AIME25	AMC23	Connect4	IFEval
Base	22.1	19.4	66.6	10.9	83.0
SPoT (Qwen3-30B-A3B Oracle)	25.6	22.5	67.0	26.9	82.9

E.2 Prompt Sensitivity of the Oracle

To verify that the gains are not artifacts of careful prompt engineering, we replace our default rectification prompt with a deliberately simplified variant:

> *"Revise a student model's answer to make it correct, while maintaining the student model's original writing style, tone, and formatting. Identify the correct parts and keep them, and replace the incorrect parts with correct reasoning. Only output the corrected answer."*

We re-curate 2k pairs with the simplified prompt and re-train under identical hyperparameters. The two prompts produce nearly identical performance across all benchmarks (Tab. 12), indicating that the framework's effectiveness is not dependent on a specific prompt formulation.

Table 12: Prompt sensitivity of the Oracle on Qwen3-8B (2k pairs). Performance is robust to prompt variation.

Prompt	AIME24	AIME25	AMC23	Connect4	IFEval
Default	26.0	19.4	67.0	26.0	84.1
Simplified	26.5	18.8	66.6	27.0	84.1

E.3 Rationale for the LCS Filtering Threshold

The LCS-based filter governs an intrinsic trade-off between curation efficiency and the on-policy constraint:

- A *strict* threshold (e.g., 0.5) yields more on-policy data but discards a large fraction of candidate pairs, making valid pairs more expensive to gather.
- A *loose* threshold (e.g., 0.7) retains more pairs at the risk of admitting off-policy data.

Starting from 6,171 error samples generated by Qwen3-8B, we evaluate two alternative LCS filtering thresholds, $\gamma = 0.5$ and $\gamma = 0.7$. Table 13 reports the number of retained training pairs and downstream performance for each setting.

Table 13: LCS filtering on Qwen3-8B: retained training pairs and downstream performance.

Setting	γ	Training pairs	AIME24	AIME25	AMC23	Connect4	IFEval
Base	–	–	22.1	19.4	66.6	10.9	83.0
SPoT	0.5	1,998	19.4	22.1	64.1	23.7	84.5
SPoT	0.7	3,646	26.0	22.1	67.0	27.9	82.3

Why a surface-level proxy suffices. A more rigorous measure of distributional proximity between rectified text and the student's own distribution would be the student's perplexity (PPL) on the rectified sequence. However, computing PPL requires a full forward pass per candidate. We thus adopt LCS as a cheap surface-level proxy, with the intuition that sequences whose tokens closely follow the student's sampling trajectory should also exhibit lower PPL under the student. To substantiate this, we computed the actual PPL on randomly sampled rectified pairs, partitioned by their LCS-based change ratio:

- 1k pairs with change ratio < 0.6 (more on-policy by LCS): mean PPL $= 1.60$.
- 1k pairs with change ratio > 0.6 (less on-policy by LCS): mean PPL $= 1.95$.

The relationship confirms that LCS is an effective proxy for student-side perplexity at curation time.

F CONNECT4 BENCHMARK AND POST-TRAINING

F.1 CONNECT4 TASK

Game Rules. Connect4 is a two-player game of perfect information played on a vertical grid of dimension 6×7. Players alternate turns dropping distinct pieces into one of the seven columns, where the piece occupies the lowest available row within that column due to simulated gravity. The objective is to form a contiguous line of four pieces either vertically, horizontally, or diagonally. It requires the LLM to *parse the board state*, *execute lookahead planning*, and *identify winning topologies*. With a state-space complexity of approximately 4.5×10^{12} legal positions, Connect4 offers a non-trivial, deterministic environment ideal for generating synthetic states to evaluate LLMs without data contamination.

Dynamic Dataset Construction. To strictly evaluate OOD reasoning without the risk of data contamination in static benchmarks, we leverage the GAMEBoT framework (Lin et al., 2025). While GAMEBoT supports full-game simulations, interactive evaluation is often inefficient and difficult to reproduce. Instead, we dynamically construct a static dataset of reasoning tasks rooted in `Connect4`. To ensure the model relies on symbolic reasoning, the board state is serialized into a text-based format (e.g., a list of coordinates).

The construction process involves:

- State Generation: We utilize two randomized agents to simulate gameplay, generating a diverse distribution of board states ranging from balanced positions to critical tactical scenarios.
- Filtering and Balancing: We filter out duplicate states. Furthermore, to prevent LLMs from inflating scores by consistently predicting "no winning moves" (due to label imbalance), we downsample states with empty answers, maintaining them at a maximum ratio of 20% of the dataset.
- Dataset Compilation: For efficient evaluation, we construct a final dataset comprising 500 distinct instances.

Evaluation Protocol. For each instance, LLMs are prompted to solve two problems in a query:

- *Are there any potential winning moves to form 4-in-a-row for you? Output all winning moves.*
- *Are there any potential winning moves to form 4-in-a-row for your opponent? Output all winning moves.*

Ground truth is computed using a perfect solver within GAMEBoT. We parse the model's Chain-of-Thought (CoT) to extract the final answers and verify them against the game engine's ground truth. Invalid and partially correct answers are counted as incorrect, with no partial credit.

Difficulty Analysis. In an additional diagnostic evaluation, we assess Qwen3-8B and Gemini 2.5 Pro across three levels of board complexity, defined by the number of pieces on the board: easy (≤ 12), medium (13–20), and hard (≥ 21). As shown in Table 14, accuracy decreases as the number of pieces increases for both models.

Table 14: Connect4 accuracy by board complexity (number of pieces). All entries are percentages.

Model	Easy (≤ 12)	Medium (13–20)	Hard (≥ 21)
Qwen3-8B	23.30	7.22	2.82
Gemini 2.5 Pro	97.91	88.67	73.12

Parsing Failures. In this diagnostic evaluation, the output parsing failure rates are 56.94% for Qwen3-8B and 0% for Gemini 2.5 Pro.

F.2 POST-TRAINING ON CONNECT4

To evaluate SPOT beyond mathematical training, we separately train Qwen3-8B on 1,000 Connect4 examples. The board positions in these training examples do not overlap with the 500-position evaluation set described in Appendix F.1. We use GPT-5.5 as the Oracle to rectify student responses, with ground-truth moves computed by a rule-based program. GPT-5.5 achieves 92% accuracy on Connect4. We deliberately omit the final-answer filter used for mathematics, retaining rectifications even when their final answers disagree with the rule-based ground truth. This design tests whether SPOT can tolerate mistakes in the teacher's rectifications while still improving the student's reasoning ability. We evaluate on the 500-position Connect4 benchmark described in Appendix F.1, while monitoring mathematical reasoning and instruction following with AIME24, AIME25, Math500, and IFEval.

As shown in Table 15, both methods substantially improve Connect4 accuracy over the base model. SPOT reaches 46.7%, compared with 43.9% for SFT, while retaining higher IFEval accuracy (83.3% versus 80.4%). Relative to the base model, it also improves AIME24, AIME25, and Math500 by 3.3, 1.9, and 1.6 percentage points, respectively. These results show that SPOT remains effective on game reasoning even when Oracle rectifications are used without final-answer filtering.

Table 15: Qwen3-8B trained on Connect4 examples. Mathematical benchmarks and IFEval monitor capabilities outside the training domain. All entries are accuracy in percent.

Model	Connect4	AIME24	AIME25	Math500	IFEval
Base	10.9	22.1	19.4	82.8	83.0
SFT	43.9	24.0	20.6	84.8	80.4
SPOT	46.7	25.4	21.3	84.4	83.3

G COST AND EFFICIENCY ANALYSIS

Oracle API cost. Curating the full 4k rectified pairs with Gemini 2.5 Pro (priced at $10/1M output tokens) cost approximately $278 in total and completed in roughly 40 minutes when run in parallel with student self-sampling. To eliminate reproducibility hurdles, we will release the full pipeline, including all curated training data, the curation and training codebases, model checkpoints, and evaluation scripts.

Wall-clock and GPU-hour comparison with GRPO. We compare end-to-end training cost between GRPO and SPOT on Qwen3-8B under identical hardware (8× H800 80GB) in Tab. 16. SPOT is efficient, as it requires only one rollout per sample.

Table 16: Wall-clock and GPU-hour for GRPO and SPOT on Qwen3-8B. Teacher rectification is parallel with student sampling and therefore does not appear as a separate term.

Component	GRPO	SPOT
Total time	∼13 h	∼1 h
Sampling	5.56 h	43 min
Policy training	5.12 h	16 min
Reference log-prob	2.25 h	(in training)
Teacher rectification	—	overlapped with sampling
Total GPU hours	∼104	∼8