UNI-OPD: UNIFYING ON-POLICY DISTILLATION WITH A DUAL-PERSPECTIVE RECIPE

Anonymous authors
Paper under double-blind review

ABSTRACT

On-policy distillation (OPD) has recently emerged as an effective post-training paradigm for consolidating the capabilities of specialized expert models into a single student model. Despite its empirical success, the conditions under which OPD yields reliable improvement remain poorly understood. We identify two fundamental bottlenecks that limit effective OPD: insufficient student exploration of informative states and unreliable teacher supervision for student rollouts. Building on this insight, we propose Uni-OPD, a unified OPD framework centered on a dual-perspective optimization strategy that generalizes across LLMs and MLLMs. On the student side, we adopt offline and online data balancing strategies to promote exploration of informative states during training. On the teacher side, we show that reliable supervision hinges on whether aggregated token-level guidance remains order-consistent with the outcome reward. We therefore introduce an outcome-guided margin calibration mechanism to restore this consistency between correct and incorrect trajectories. We conduct extensive experiments on 16 benchmarks across 5 domains, spanning single-teacher, multi-teacher, strong-to-weak, and cross-modal distillation settings. The results verify the effectiveness and versatility of Uni-OPD, with an average gain of **2.1** points over vanilla OPD. Code is available at https://anonymous.4open.science/r/Uni-OPD/.

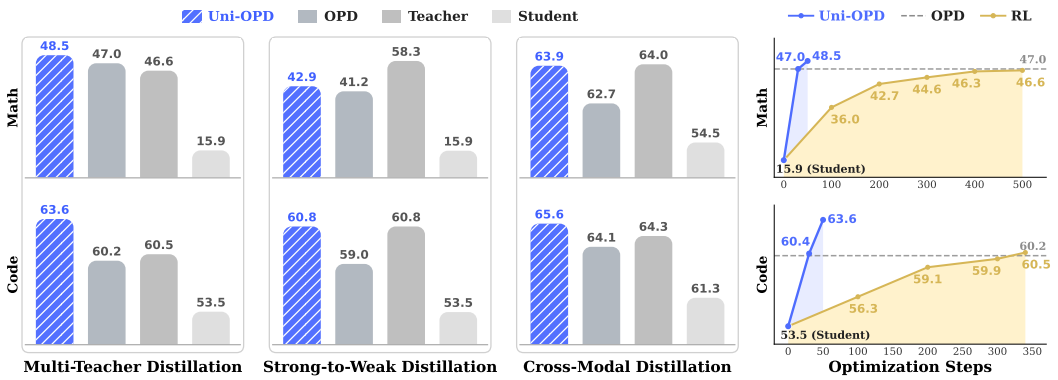


Figure 1: **Overall performance comparisons and convergence behavior.** Results are shown for settings including multi-teacher, strong-to-weak, and cross-modal distillation on math reasoning and code generation tasks. Uni-OPD consistently outperforms OPD and converges faster than RL, demonstrating its effectiveness across diverse settings.

1 INTRODUCTION

Injecting complex reasoning abilities, domain knowledge, and human preferences into LLMs and MLLMs remains a core challenge in the post-training stage. Conventional approaches typically follow a two-stage paradigm: supervised fine-tuning (SFT) first, followed by reinforcement learning (RL) (Guo et al., 2025a; Zeng et al., 2026; Zhao et al., 2026a). While SFT leverages expert data for training, its inherently off-policy nature introduces substantial exposure bias (Qin et al., 2024; Song & Zheng, 2026). Entering rarely covered erroneous states during inference may lead to compounding errors. Alternatively, on-policy RL (*e.g.,* GRPO (Shao et al., 2024b)) alleviates distribution shift through online sampling. However, it mainly relies on sequence-level or terminal rewards, making fine-grained credit assignment difficult and limiting long-term training stability (Team et al., 2026).

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.

Recently, on-policy distillation (OPD) has emerged as a promising post-training paradigm for efficiently transferring the knowledge of domain experts into a single model. It combines the strengths of RL and SFT: on-policy sampling and token-level supervision. OPD trains the student on its own trajectories with teacher feedback under a reverse KL objective (Lu & Lab, 2025; Xu et al., 2026a).

Despite its empirical success, OPD research has primarily focused on text-only LLM distillation (Zhou et al., 2025; Ye et al., 2025; Yang et al., 2026b; Xiao et al., 2026; Yang et al., 2026c; Wu et al., 2026). Some works have extended OPD to MLLMs, yet target specific settings such as vision-language, video, or speech (Bousselham et al., 2025; Ko et al., 2026; Li et al., 2026b; Cao et al., 2026). We therefore develop a unified OPD framework for both LLMs and MLLMs, enabling effective knowledge distillation across tasks and modalities.

Key observations. Beyond unifying the framework, we raise a more fundamental question: *what makes OPD a reliable optimization paradigm?* We posit that effective OPD depends on two factors. **First**, the student must sufficiently explore informative states, *i.e.,* diverse and appropriately difficult self-generated trajectories. **Second**, the teacher's token-level supervision must remain reliable when applied to student rollouts. In particular, the reliability of token-level guidance is enhanced when its trajectory-level aggregation remains order-consistent with outcome reward (*i.e.,* correct trajectories receive higher aggregated scores than incorrect ones). The outcome reward thus provides a global anchor for calibrating unreliable teacher supervision. These observations motivate a dual-perspective optimization strategy that jointly improves student exploration and teacher reliability.

Our recipe. Building on these insights, we introduce `Uni-OPD`, a dual-perspective strategy for optimizing OPD from the fundamental roles of the student and the teacher. In this unified framework, we adopt two complementary data-balancing strategies, namely offline difficulty-aware balancing and online correctness-aware balancing, to promote exploration of informative student-generated states. We further present a novel online outcome-guided margin calibration mechanism to obtain reliable teacher supervision. As shown in Fig. 1, extensive experiments across 3 distillation settings demonstrate that `Uni-OPD` consistently outperforms OPD and converges faster than RL.

To summarize, our contributions are threefold:

- **Key bottlenecks of OPD.** We identify two core bottlenecks in OPD: insufficient exploration of informative states and unreliable teacher supervision for student rollouts. Our analysis reveals that reliable teacher supervision largely depends on whether token-level guidance remains order-consistent with the outcome reward.
- **Dual-perspective optimization recipe.** We present a dual-perspective optimization recipe for unified OPD that jointly improves student exploration and teacher supervision during training. Concretely, we combine offline and online data balancing strategies with an outcome-guided margin calibration mechanism, leading to more effective optimization.
- **Comprehensive experimental validation.** We conduct extensive experiments on 5 domains and 16 benchmarks covering diverse settings, including single-teacher and multi-teacher distillation across LLMs and MLLMs, strong-to-weak distillation, and cross-modal distillation (*i.e.,* combining text-only and multimodal tasks). Our results verify the effectiveness and versatility of `Uni-OPD` (**+2.1** points over vanilla OPD on average) and provide insights into reliable OPD.

2 Related Work

Knowledge distillation for LLMs and MLLMs. Knowledge distillation (Hinton et al., 2015; Xu et al., 2024) aims to transfer knowledge from a larger teacher model to a smaller student model. Conventional approaches typically rely on off-policy forward Kullback–Leibler (KL) divergence on a static dataset to align the generation distribution of student with that of the teacher (Liu et al., 2024d; Guo et al., 2025b; He et al., 2025a; Liu & Zhang, 2025; Ko et al., 2025). Another line of work uses SFT on tokens generated by the teacher as an alternative off-policy distillation strategy for eliciting reasoning capabilities during post-training (Guo et al., 2025a; Zhang et al., 2025c; Bansal et al., 2026; Zhang et al., 2025b; Team et al., 2026; Xiao et al., 2026). Though effective, these off-policy methods essentially imitate the behavior of teacher, limiting the ability of student policy to surpass the teacher and making the student prone to exposure bias (Song & Zheng, 2026).

On-policy distillation. OPD (Agarwal et al., 2024; Lu & Lab, 2025) allows a superior teacher to provide feedback on the on-policy trajectories of student. This paradigm effectively alleviates expo-

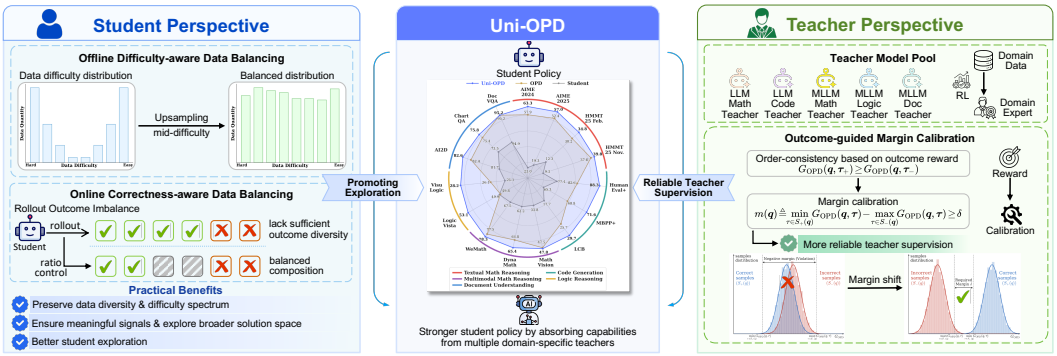


Figure 2: **Overview of the `Uni-OPD` framework.** (*Left*) Offline difficulty-aware and online correctness-aware data balancing promote student exploration. (*Right*) Outcome-guided margin calibration mechanism improves the reliability of teacher supervision. (*Middle*) The resulting student policy merges complementary capabilities from multiple domain-specific teachers more effectively than standard OPD, leading to stronger overall performance.

sure bias and elevates the student's upper performance bound. Owing to these merits, OPD has become an efficient way to merge capabilities from multiple experts into a single student model (Xiao et al., 2026; Yang et al., 2026c), as well as to support strong-to-weak distillation (Bai et al., 2025a; Zeng et al., 2026). Building on this paradigm, current studies on OPD have branched into several key directions. From the lens of the teacher, recent work explores self-distillation paradigms (Kujanpää et al., 2024; Shenfeld et al., 2026; Zhao et al., 2026b; Hübotter et al., 2026; Ye et al., 2026; Zhang et al., 2026a; Stein et al., 2026), develops black-box OPD methods (Ye et al., 2025; Xiong et al., 2026), and facilitates distillation across different model families (Patiño et al., 2025). Complementary efforts focus on training frameworks (Zhang et al., 2026b) and stable optimization strategies (Jin et al., 2026; Kim & Baek, 2026; Li et al., 2026c; Xu et al., 2026b) combined with RL (Yang et al., 2026a; Qu et al., 2026; Jang et al., 2026; Wang et al., 2026). Few works extend OPD to multimodal domains (Bousselham et al., 2025; Ko et al., 2026; Li et al., 2026b; Cao et al., 2026). In this work, we push OPD with a dual-perspective recipe that promotes student exploration and teacher reliability, generalizing across LLMs and MLLMs. Detailed related work is provided in §F.

3 Methodology

We propose `Uni-OPD`, a unified framework that advances OPD across LLMs and MLLMs. As shown in Fig. 2, our design is driven by two bottlenecks in OPD: *insufficient exploration of informative student-generated states* and *unreliable teacher supervision for student rollouts*. `Uni-OPD` addresses them with a dual-perspective recipe that enhances student exploration and calibrates teacher supervision to align with the outcome reward. In this section, we describe our method in detail.

3.1 Preliminaries

On-policy distillation. In OPD, the student policy $\pi_{\boldsymbol{\theta}}$ is trained on its own sampled trajectories by minimizing the reverse Kullback–Leibler (KL) divergence to the teacher policy π_{T}:

$$\mathcal{L}_{\mathrm{OPD}}(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{q}\sim\mathcal{D}}\left[\mathcal{D}_{\mathrm{KL}}\left(\pi_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{q}) \| \pi_{\mathrm{T}}(\cdot \mid \boldsymbol{q})\right)\right], \quad \boldsymbol{\theta}^{\star} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}_{\mathrm{OPD}}(\boldsymbol{\theta}). \tag{1}$$

where $\boldsymbol{q}$ is the input question. The exact policy gradient involves the cumulative future log-ratio terms. In practice, following G-OPD (Yang et al., 2026b), existing OPD implementations use a zero discount factor and adopt the following next-token approximation:

$$\nabla_{\boldsymbol{\theta}}\mathcal{L}_{\mathrm{OPD}}(\boldsymbol{\theta}) \approx \mathbb{E}_{\boldsymbol{q}\sim\mathcal{D},\,\boldsymbol{\tau}\sim\pi_{\boldsymbol{\theta}}(\cdot|\boldsymbol{q})}\left[\sum_{t=1}^{|\boldsymbol{\tau}|}\left(\log \pi_{\boldsymbol{\theta}}(o_t \mid \boldsymbol{q}, \boldsymbol{o}_{<t}) - \log \pi_{\mathrm{T}}(o_t \mid \boldsymbol{q}, \boldsymbol{o}_{<t})\right)\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(o_t \mid \boldsymbol{q}, \boldsymbol{o}_{<t})\right], \tag{2}$$

where $\boldsymbol{\tau} = (o_1, \ldots, o_{|\boldsymbol{\tau}|})$ is a trajectory sampled by the student, o_t is the token at step t, and $|\boldsymbol{\tau}|$ is the length of the trajectory; $\boldsymbol{o}_{<t}$ denotes the prefix before step t. The gradient naturally induces a token-level reward at step t, analogous to standard RL:

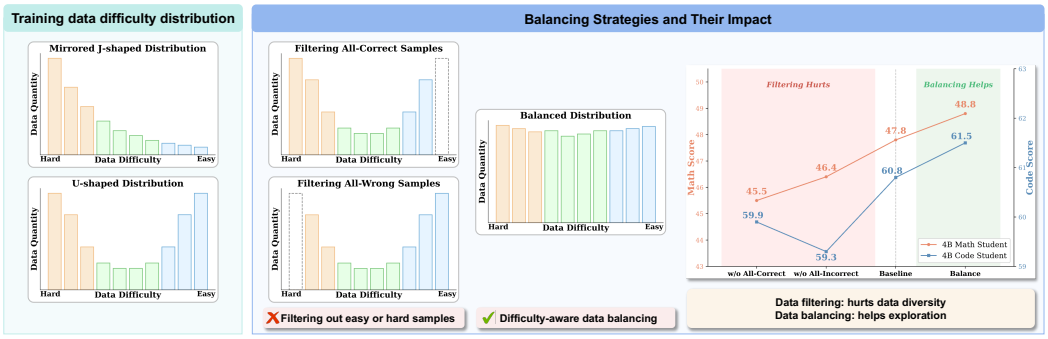


Figure 3: **Data difficulty distribution and its impact on OPD performance.** (*Left*) Training data often exhibits mirrored J-shaped or U-shaped difficulty distributions. (*Right*) A naive strategy is to filter out overly easy or overly hard samples (*i.e.,* all-correct or all-wrong cases), but this reduces diversity. In contrast, our difficulty-balancing strategy upsamples mid-difficulty samples to preserve a balanced spectrum and empirically outperforms filtering.

$$r_t^{\text{OPD}} = \log \pi_{\text{T}}(o_t \mid \boldsymbol{q}, \boldsymbol{o}_{<t}) - \log \pi_{\boldsymbol{\theta}}(o_t \mid \boldsymbol{q}, \boldsymbol{o}_{<t}) = \log \frac{\pi_{\text{T}}(o_t \mid \boldsymbol{q}, \boldsymbol{o}_{<t})}{\pi_{\boldsymbol{\theta}}(o_t \mid \boldsymbol{q}, \boldsymbol{o}_{<t})}. \tag{3}$$

This formulation provides fine-grained credit assignment signals at the token level. However, such supervision is not always reliable in practice (see §B.1 for an analysis of these failure modes).

3.2 The Overview of Uni-OPD

In this work, we propose `Uni-OPD`, a unified OPD framework that generalizes across both LLMs and MLLMs, as illustrated in Fig. 2. Formally, given M teachers $\{\pi_{\text{T}_1}, \pi_{\text{T}_2}, \ldots, \pi_{\text{T}_M}\}$ who specialize in different domains, let $w_i(\boldsymbol{q})$ denote the weight assigned to teacher π_{T_i} for prompt $\boldsymbol{q}$:

$$\mathcal{L}_{\text{Uni-OPD}}(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{q}\sim\mathcal{D}}\left[\sum_{i=1}^{M} w_i(\boldsymbol{q})\, \mathcal{D}_{\text{KL}}\left(\pi_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{q}) \| \pi_{\text{T}_i}(\cdot \mid \boldsymbol{q})\right)\right], \qquad \sum_{i=1}^{M} w_i(\boldsymbol{q}) = 1. \tag{4}$$

This formulation provides a unified objective for both single-teacher and multi-teacher distillation by aggregating supervision from multiple experts. Building on this objective, we optimize OPD from the two fundamental roles. On the student side, we introduce a data-balancing strategy that promotes exploration via offline difficulty-aware and online correctness-aware selection. On the teacher side, we develop an outcome-guided margin calibration mechanism to correct unreliable token-level supervision by enforcing consistency with outcome rewards.

3.3 Joint Data Balancing for Student Exploration

Offline difficulty-aware data balancing. A prevalent practice in RL is to estimate prompt difficulty via multiple rollouts and then filter out samples that are either overly easy (*i.e.,* always correct) or overly hard (*i.e.,* always incorrect) (An et al., 2025; Zhou et al., 2023a). However, for small-scale models, training data often exhibits a mirrored J-shaped or U-shaped distribution (see Fig. 3). Strictly removing these easy or hard samples can substantially reduce data diversity and limit exploration of informative student-generated states. Our empirical findings show that such filtering leads to substantial performance degradation in OPD.

Based on this observation, we adopt a difficulty-aware balancing strategy that selectively upsamples the under-represented difficulty tiers of each data source (§A.1). As shown in Fig. 3, this strategy reshapes the data distribution into a more uniform form while preserving both diversity and difficulty. In addition, it consistently improves performance on math reasoning and code generation. Overall, these results show that maintaining data diversity and a balanced difficulty spectrum enables the student to generate more informative trajectories, thereby exploring a broader solution space.

Online correctness-aware data balancing. After applying offline difficulty-aware balancing, we further observe that insufficient exploration can cause the model to collapse to local optima

during training, especially when rollout groups lack sufficient outcome diversity (*e.g.,* only incorrect trajectories). To mitigate this issue, we enforce a balanced composition of correct and incorrect trajectories within each mixed rollout group. Degenerate groups where all samples share the same outcome retain their original OPD rewards and do not undergo margin calibration. By maintaining this balance in mixed groups, the student receives meaningful contrastive signals for stable on-policy learning. As shown in Fig. 4, an outcome balance achieves higher scores than using only correct samples or an excessively high correct/incorrect ratio.

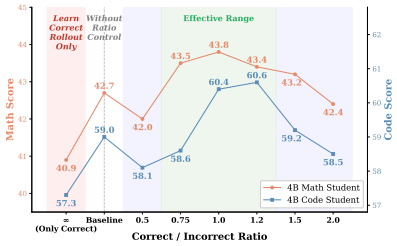


Figure 4: Impact of online correct and incorrect ratio on student scores.

3.4 Outcome-guided Calibration for Teacher Supervision

A basic premise of OPD is that the teacher exhibits a directional likelihood preference over positive and negative trajectories. In particular, relative to the student, the teacher should assign higher likelihood to correct trajectories and lower likelihood to incorrect ones. Under this premise, the resulting distillation signal should remain consistent with outcome-level correctness at the trajectory level. We next formalize this principle through a trajectory-level distillation return and develop an outcome-guided calibration strategy based on it.

Trajectory-level distillation return. To characterize the overall supervision signal along a rollout trajectory, we define the *trajectory-level distillation return* as the average log-probability gap between the teacher and the student:

$$G_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau}) \triangleq \frac{1}{|\boldsymbol{\tau}|}\sum_{t=1}^{|\boldsymbol{\tau}|}\log\frac{\pi_{\mathrm{T}}(o_t \mid \boldsymbol{q},\boldsymbol{o}_{<t})}{\pi_{\boldsymbol{\theta}}(o_t \mid \boldsymbol{q},\boldsymbol{o}_{<t})} = \frac{1}{|\boldsymbol{\tau}|}\sum_{t=1}^{|\boldsymbol{\tau}|} r_t^{\mathrm{OPD}}, \tag{5}$$

This quantity measures the teacher's *average log-likelihood preference* over the student along trajectory $\boldsymbol{\tau}$. When $G_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau}) > 0$, the teacher assigns higher confidence than the student on average, encouraging the student to move toward this trajectory. Conversely, when $G_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau}) < 0$, the student is discouraged from moving toward this trajectory. The normalization by trajectory length ensures comparability across trajectories of different lengths.

Order consistency as a trajectory-level criterion. For a given question $\boldsymbol{q}$, let $R(\boldsymbol{q},\boldsymbol{\tau}) \in \{0,1\}$ denote the outcome reward of a sampled trajectory $\boldsymbol{\tau}$, where $R(\boldsymbol{q},\boldsymbol{\tau}) = 1$ indicates that the final answer in $\boldsymbol{\tau}$ is correct for question $\boldsymbol{q}$. We then define the positive and negative trajectory sets as:

$$S_+(\boldsymbol{q}) \triangleq \{\boldsymbol{\tau} \mid R(\boldsymbol{q},\boldsymbol{\tau}) = 1\}, \qquad S_-(\boldsymbol{q}) \triangleq \{\boldsymbol{\tau} \mid R(\boldsymbol{q},\boldsymbol{\tau}) = 0\}. \tag{6}$$

Following the trajectory-level bandit formulation in Ouyang et al. (2022), we treat the prompt as the context and the entire generated trajectory as a macro-action. Under this view, the associated outcome reward naturally serves as a one-step trajectory-level return, denoted as $G_{\mathrm{RL}}(\boldsymbol{q},\boldsymbol{\tau}) = R(\boldsymbol{q},\boldsymbol{\tau})$. Therefore, the outcome-level RL return induces the following oracle ordering:

$$G_{\mathrm{RL}}(\boldsymbol{q},\boldsymbol{\tau}_+) \geq G_{\mathrm{RL}}(\boldsymbol{q},\boldsymbol{\tau}_-), \qquad \forall \boldsymbol{\tau}_+ \in S_+(\boldsymbol{q}),\ \forall \boldsymbol{\tau}_- \in S_-(\boldsymbol{q}). \tag{7}$$

The derivation is provided in §B.2. This motivates a trajectory-level reliability criterion for OPD. Under the distillation premise, the trajectory-level distillation return $G_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau})$ should preserve the same outcome-induced ordering as $G_{\mathrm{RL}}(\boldsymbol{q},\boldsymbol{\tau})$. Specifically, for any prompt $\boldsymbol{q}$, we expect:

$$G_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau}_+) \geq G_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau}_-), \qquad \forall \boldsymbol{\tau}_+ \in S_+(\boldsymbol{q}),\ \forall \boldsymbol{\tau}_- \in S_-(\boldsymbol{q}). \tag{8}$$

Teacher supervision may violate ordering. In practice, however, the teacher's supervision is not always reliable. As discussed in §B.1, teacher scoring may degrade in sparse out-of-distribution regions, overestimate incorrect trajectories, or underestimate correct ones due to spurious local patterns. Such failures may persist even after token-level supervision is aggregated to the trajectory level. Averaging alone therefore cannot certify reliability, since the mismatch is often concentrated in a few extreme samples: a single overly confident negative trajectory or a severely underestimated positive trajectory can already distort the supervision signal for the entire rollout group.

Outcome-guided margin calibration. Based on the above analysis, during OPD training, the constraint in Eq. (8) should hold between positive and negative trajectories within each rollout group.

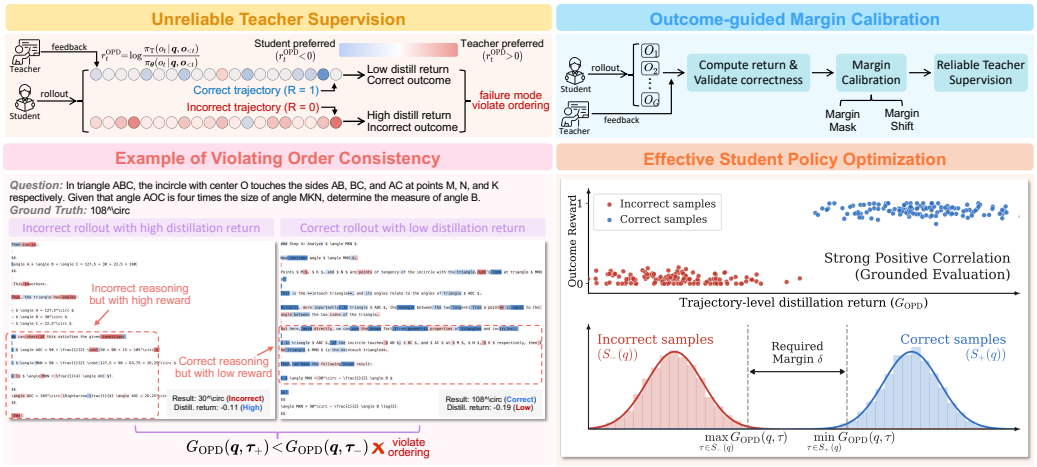


Figure 5: **Demonstration of unreliable teacher supervision and outcome-guided margin calibration mechanism.** (*Left*) Vanilla teacher supervision in OPD suffers from misalignment between trajectory-level return and outcome rewards, yielding unreliable supervision. (*Right*) Our method uses outcome rewards as a global anchor to calibrate returns through margin-based adjustment.

To this end, we consider the margin between the lowest-scoring correct trajectory and the highest-scoring incorrect trajectory, which directly characterizes whether the ordering is violated in the most adversarial case. We define the prompt-level margin as

$$m(\boldsymbol{q}) \triangleq \min_{\boldsymbol{\tau} \in S_+(\boldsymbol{q})} G_{\mathrm{OPD}}(\boldsymbol{q}, \boldsymbol{\tau}) - \max_{\boldsymbol{\tau} \in S_-(\boldsymbol{q})} G_{\mathrm{OPD}}(\boldsymbol{q}, \boldsymbol{\tau}) \,. \tag{9}$$

By construction, $m(\boldsymbol{q}) \geq 0$ indicates order consistency on prompt $\boldsymbol{q}$, since even the worst positive trajectory still does not score below the best negative one (see Fig. 5). For robustness, we require:

$$m(\boldsymbol{q}) \geq \delta \,, \tag{10}$$

where $\delta > 0$ defines a safety margin against estimation noise and finite-sample fluctuations. Since $S_+(\boldsymbol{q})$ and $S_-(\boldsymbol{q})$ are determined by outcome rewards, this criterion uses the outcome signal as a global anchor to calibrate the teacher's trajectory-level scores. This formulation enables direct interventions on the margin, allowing us to suppress ordering violations or enlarge the separation between positive and negative trajectories.

Margin calibration strategy. Based on Eq. (10), we present two calibration strategies: **margin mask** and **margin shift**. The Margin mask greedily drops the most ordering-violating trajectories until $m(\boldsymbol{q}) \geq \delta$ holds, so that training is performed only with reliable supervision. Margin shift instead repairs an unreliable group with the smallest additive correction. For $m(\boldsymbol{q}) < \delta$, we define:

$$\lambda(\boldsymbol{q}) = \delta - m(\boldsymbol{q}), \qquad \widetilde{G}_{\mathrm{OPD}}(\boldsymbol{q}, \boldsymbol{\tau}) = G_{\mathrm{OPD}}(\boldsymbol{q}, \boldsymbol{\tau}) + \lambda(\boldsymbol{q})\, \mathbf{1}\{R(\boldsymbol{q}, \boldsymbol{\tau}) = 1\}. \tag{11}$$

This shift preserves the relative ordering within $S_+(\boldsymbol{q})$ and guarantees

$$\min_{\boldsymbol{\tau} \in S_+(\boldsymbol{q})} \widetilde{G}_{\mathrm{OPD}}(\boldsymbol{q}, \boldsymbol{\tau}) - \max_{\boldsymbol{\tau} \in S_-(\boldsymbol{q})} \widetilde{G}_{\mathrm{OPD}}(\boldsymbol{q}, \boldsymbol{\tau}) = \delta \,. \tag{12}$$

In this way, margin shift restores outcome-consistent ordering with a minimal group-level correction, while margin mask provides a more conservative alternative when the supervision signal is too unreliable to calibrate. Detailed description and analysis are provided in §B.

4 Experiments and Analysis

In this section, we conduct comprehensive experiments to evaluate the effectiveness of `Uni-OPD`. We assess how the proposed recipe improves OPD performance across diverse scenarios for LLMs and MLLMs, including single-teacher and multi-teacher distillation (§4.1), strong-to-weak distillation (§4.2), and cross-modal distillation (§4.3). Then, we provide a rigorous ablation study and qualitative analysis to demonstrate the core strategies of our method (§B.6, §4.4, §E.3, §G).

Due to the page limit, we detail the *Experimental Setup* in the appendix, including models (§C.1), training data (§C.3), evaluation benchmarks (§D.1), and baselines (§D.2).

Table 1: **Performance of *Qwen3-4B Student* under math reasoning and code generation benchmarks.** Teacher models (*i.e., Qwen3-4B-Math-RL* and *Qwen3-4B-Code-RL*) are developed through domain-specific RL. The performance of teacher models is denoted by the "RL" type. **Bold** values indicate the best score within each group. **Avg.** denotes the average score within each domain.

Method	Math Reasoning					Code Generation			
	AIME 2024	AIME 2025	HMMT 25 Feb.	HMMT 25 Nov.	Avg.	Human Eval+	MBPP+	LCB	Avg.
Student (4B)	23.0	19.3	12.3	9.2	15.9	77.4	65.3	17.7	53.5
Teacher (RL)	60.1	55.1	32.5	38.5	46.6	85.2	69.8	26.6	60.5
Single–Teacher Distillation									
ExPO	58.7	55.2	32.4	37.0	45.8	84.8	70.2	28.0	61.0
OPD	57.9	52.4	30.2	37.8	44.6	82.6	68.8	25.7	59.0
ExOPD	62.7	56.1	33.9	39.3	48.0	86.9	70.7	28.6	62.1
`Uni-OPD`	**63.3**	**57.0**	**34.8**	**39.8**	**48.7**	**88.3**	**71.6**	**29.7**	**63.2**
Multi–Teacher Distillation									
SFT	58.5	53.3	30.7	34.8	44.3	86.4	69.6	26.4	60.8
ExPO	57.5	54.5	31.7	36.3	45.0	86.7	72.0	29.0	62.6
OPD	60.9	55.2	33.4	38.3	47.0	86.3	70.9	23.4	60.2
ExOPD	61.0	56.0	34.4	39.2	47.7	86.3	70.6	29.0	62.0
`Uni-OPD`	**62.3**	**57.2**	**34.9**	**39.6**	**48.5**	**88.0**	**72.6**	**30.1**	**63.6**

Table 2: **Performance of *Qwen3-VL-4B-Instruct Student* on math reasoning, logic reasoning, and document understanding benchmarks.** *Qwen3-VL-4B-Instruct-Math-RL*, *Qwen3-VL-4B-Instruct-Logic-RL*, and *Qwen3-VL-4B-Instruct-Document-RL* are the teacher models.

Method	Math Reasoning				Logic Reasoning				Document Understanding				
	Math Vision	Dyna Math	We Math	Avg.	LogicVista Accuracy	LogicVista Format	Visu Logic	Avg.	AI2D	Chart QA	Doc VQA	Info VQA	Avg.
Student (4B)	33.8	62.2	67.5	54.5	49.9	66.4	25.1	47.1	81.7	73.5	94.9	79.8	82.5
Teacher (RL)	47.2	65.3	79.5	64.0	52.5	73.8	27.4	51.2	82.5	76.4	95.1	81.6	83.9
Single–Teacher Distillation													
OPD	47.5	64.8	77.5	63.3	49.8	73.0	26.1	49.6	82.4	75.4	95.2	**81.4**	83.6
`Uni-OPD`	**47.8**	**65.4**	**78.3**	**63.9**	**53.1**	**73.8**	**28.2**	**51.7**	**82.6**	**75.8**	**95.2**	81.2	**83.7**
Multi–Teacher Distillation													
OPD	41.0	60.9	71.7	57.9	51.3	72.3	26.3	50.0	82.6	75.0	95.1	81.3	83.4
`Uni-OPD`	**45.5**	**62.3**	**76.1**	**61.3**	**54.0**	**75.2**	**27.5**	**52.2**	**83.0**	**75.7**	**95.3**	**81.6**	**83.9**

4.1 Single-Teacher and Multi-Teacher Distillation

As an effective paradigm for consolidating capabilities from one or more teachers into a unified student model, we evaluate `Uni-OPD` on LLMs and MLLMs across diverse domains. For LLMs, following G-OPD (Yang et al., 2026b), we conduct experiments on math reasoning and code generation. For MLLMs, we consider math reasoning, logic reasoning, and document understanding.

Main results. As shown in Table 1, `Uni-OPD` achieves the best overall performance on LLM distillation under both single-teacher and multi-teacher settings, consistently surpassing OPD and ExOPD. It reaches 48.7 on math reasoning and 63.2 on code generation with a single teacher, and improves over OPD by 1.5 and 3.4 points when multiple teachers are merged into one student.

A similar trend holds for MLLMs in Table 2. Under single-teacher distillation, `Uni-OPD` achieves the best averages in all three domains: 63.9 on math reasoning, 51.7 on logic reasoning, and 83.7 on document understanding. With multiple teachers, it improves OPD from 57.9, 50.0, and 83.4 to 61.3, 52.2, and 83.9, respectively, validating its robustness across settings. To further validate the effectiveness of `Uni-OPD` across model scales, we provide additional results for single- and multi-teacher distillation in Tables E.1 and E.2.

4.2 Strong-to-Weak Distillation

Strong-to-weak distillation is particularly important for the practical post-training of small models (Bai et al., 2025a). We further investigate whether `Uni-OPD` can better facilitate the transfer of reasoning capabilities from a larger, stronger teacher model (*e.g.,* Qwen3-30B-A3B-Instruct-2507) to significantly smaller students (*e.g.,* Qwen3-4B and Qwen3-1.7B). In this setting, the student is trained on both math and code data, with teacher feedback provided across both domains.

Table 3: **Results for strong-to-weak distillation setting under math reasoning and code generation benchmarks.** The teacher model is *Qwen3-30B-A3B-Instruct-2507*, and the student models are the smaller *Qwen3-4B* and *Qwen3-1.7B*. "–" denotes results not reported in the original paper.

Method	Math Reasoning					Code Generation			
	AIME 2024	AIME 2025	HMMT 25 Feb.	HMMT 25 Nov.	**Avg.**	Human Eval+	MBPP+	LCB	**Avg.**
Teacher	72.1	61.4	42.5	57.1	58.3	81.9	77.2	23.4	60.8
Qwen3-4B Student									
Student	23.0	19.3	12.3	9.2	15.9	77.4	65.3	17.7	53.5
OPD	56.5	46.4	28.5	33.4	41.2	82.9	**72.4**	21.6	59.0
IW-OPD	**57.5**	49.7	28.7	–	–	78.7	70.9	–	–
`Uni-OPD`	55.9	**50.2**	**29.8**	**35.6**	**42.9**	**83.1**	71.3	**28.0**	**60.8**
Qwen3-1.7B Student									
Student	13.9	11.1	5.6	4.9	8.9	61.9	53.4	11.9	42.4
OPD	**35.7**	27.6	17.2	14.6	23.8	67.1	56.7	23.4	49.1
IW-OPD	35.5	29.5	16.4	–	–	65.2	55.0	–	–
`Uni-OPD`	35.2	**30.7**	**17.7**	**16.4**	**25.0**	**71.5**	**58.6**	**28.0**	**52.7**

Main results. The strong-to-weak results are presented in Table 3. `Uni-OPD` yields significant gains for both 4B and 1.7B students. With the 30B teacher, `Uni-OPD` consistently outperforms standard OPD. For the 4B student, `Uni-OPD` achieves average scores of 42.9 in mathematical reasoning and 60.8 in code generation, surpassing OPD by 1.7 and 1.8 points, respectively. For the smaller 1.7B student, `Uni-OPD` reaches 25.0 on math reasoning and 52.7 on code generation. These results demonstrate that `Uni-OPD` bridges the capacity gap, enabling smaller students to absorb and replicate complex reasoning behaviors from superior teachers.

4.3 Cross-Modal Distillation

Table 4: **Results for cross-modal distillation under textual code generation and multimodal math reasoning benchmarks.** The student model is *Qwen3-VL-4B-Instruct*. The teacher models are developed from the same MLLM via domain-specific RL on textual code and multimodal math domains, *i.e., Qwen3-VL-4B-Instruct-Code-RL* and *Qwen3-VL-4B-Instruct-Math-RL*, respectively.

Method	Code Generation (Textual)				Math Reasoning (Multimodal)			
	Human Eval+	MBPP+	LCB	**Avg.**	Math Vision	Dyna Math	We Math	**Avg.**
Student (4B)	76.8	70.0	37.0	61.3	33.8	62.2	67.5	54.5
Teacher (RL)	82.2	70.5	40.1	64.3	47.2	65.3	79.5	64.0
OPD	83.1	70.6	38.6	64.1	46.1	65.4	76.6	62.7
`Uni-OPD`	**84.1**	**71.4**	**41.3**	**65.6**	**46.6**	**66.5**	**78.5**	**63.9**

Cross-modal distillation is an interesting setting in OPD. Unlike general distillation settings, where capability transfer typically occurs within the same modality, here **we investigate whether textual and multimodal capabilities can be unified into a single student policy**. Specifically, we use *Qwen3-VL-4B-Instruct* as the student model, and construct domain-specific teachers from the same MLLM via RL on textual code data and multimodal math data, respectively. As a result, although the student is multimodal, one of the transferred capabilities is learned from a teacher specialized in a purely textual domain, enabling capability transfer across modality boundaries.

Main results. As shown in Table 4, `Uni-OPD` achieves consistent gains over standard OPD across both textual code generation and multimodal math reasoning in this cross-modal setting. Specifically, it improves the average score from 64.1 to 65.6 on code generation and from 62.7 to 63.9 on math reasoning. On the textual side, the gains are consistent across all three code benchmarks, with the largest improvement on LCB (38.6 $\rightarrow$ 41.3). On the multimodal side, `Uni-OPD` further improves MathVision (46.1 $\rightarrow$ 46.6) and DynaMath (65.4 $\rightarrow$ 66.5), while maintaining strong performance on WeMath. These results suggest that `Uni-OPD` can effectively absorb and coordinate capabilities originating from both textual and multimodal domains within a unified student model, rather than improving one domain at the expense of the other. For a broader evaluation of cross-modal distillation, we further report results for code and logic reasoning in §E (Table E.3).

4.4 Ablation Study

Table 5: **Results of `Uni-OPD` variants with a *Qwen3-4B Student* on math reasoning and code generation.** We ablate core strategies to assess their effectiveness using the *Qwen3-4B-RL* and *Qwen3-30B-A3B-Instruct-2507* teacher models.

Configuration	Math Reasoning					Code Generation			
	AIME 2024	AIME 2025	HMMT 25 Feb.	HMMT 25 Nov.	**Avg.**	Human Eval+	MBPP+	LCB	**Avg.**
Qwen3-4B RL Teacher									
OPD	60.9	55.2	33.4	38.3	47.0	86.3	70.9	23.4	60.2
`Uni-OPD`	62.3	57.2	34.9	39.6	**48.5**	88.0	72.6	30.1	**63.6**
w/o offline data balancing	62.6	56.5	32.5	38.5	47.5	88.0	71.1	27.9	62.3
w/o online data balancing	62.5	56.7	33.2	38.9	47.8	88.0	71.8	28.0	62.6
w/o margin calibration	63.0	54.7	33.4	38.1	47.3	86.4	71.6	25.7	61.2
Qwen3-30B-A3B-Instruct-2507 Teacher									
OPD	56.5	46.4	28.5	33.4	41.2	82.9	72.4	21.6	59.0
`Uni-OPD`	55.9	50.2	29.8	35.6	**42.9**	83.1	71.3	28.0	**60.8**
w/o offline data balancing	57.1	46.3	28.8	36.8	42.2	80.6	70.3	28.0	59.6
w/o online data balancing	57.0	47.6	26.8	37.0	42.1	81.6	71.4	28.0	60.3
w/o margin calibration	54.9	48.1	29.1	35.8	42.0	82.8	70.4	25.7	59.6

In Table 5, we conduct comprehensive ablation studies to evaluate the individual contributions of each strategy in our `Uni-OPD`. Applying our proposed operations results in a significant improvement in accuracy over the vanilla OPD. In particular, the average gains reach +1.5/+3.4 points on math/code with the Qwen3-4B-RL teacher, and +1.7/+1.8 points with the Qwen3-30B-A3B-Instruct teacher. **Offline and online data balancing address insufficient exploration**: without either of them, the student policy struggles to be exposed to diverse and challenging trajectories. **Margin calibration improves supervision reliability**: without it, token-level feedback can become misaligned with outcome rewards, leading to less stable training and suboptimal performance.

Margin mask vs. margin shift. We consider various strategies to calibrate the return signals for improving teacher supervision. In this work, we explore two simple variants, namely margin mask and margin shift. As shown in Table 6, directly incorporating either mechanism into OPD yields consistent performance gains over the baseline, underscoring the necessity of reliable teacher supervision. Among them, margin shift achieves slightly better results and is therefore adopted in our main experiments. More ablations, a token-level visualization of the calibration effect, and a summary of findings across settings are provided in §B.6, §E.3, §E.4.

Table 6: **Comparison results for different margin calibration.** We directly incorporate them into OPD to examine which strategy better benefits OPD training.

Method	AIME 2024	AIME 2025	HMMT 25 Feb.	HMMT 25 Nov.	**Avg.**
Student (4B)	23.0	19.3	12.3	9.2	15.9
OPD	57.9	52.4	30.2	37.8	44.6
+ margin mask	62.3	56.2	34.3	38.1	47.7
+ margin shift	62.7	56.3	34.4	39.2	48.2

5 Conclusion and Future Work

In this work, we present `Uni-OPD`, a unified OPD framework that generalizes across LLMs and MLLMs. We identify two key bottlenecks for effective OPD: insufficient student exploration of informative states and unreliable teacher supervision for student rollouts. To address them, we propose a dual-perspective optimization strategy: (i) offline difficulty-aware and online correctness-aware data balancing strategies for student exploration, and (ii) an outcome-guided margin calibration mechanism for teacher supervision. Extensive experiments on 16 benchmarks covering multi-teacher, strong-to-weak, and cross-modal settings demonstrate the effectiveness and versatility of `Uni-OPD`. These findings further suggest that teacher value comes from the capability gap rather than absolute strength alone, and that OPD can jointly consolidate textual and multimodal capabilities in a single student. We hope it can provide a practical foundation for future research on scalable and reliable distillation across models and modalities.

For future work, our findings suggest several promising directions: (a) extending `Uni-OPD` to larger-scale teacher distillation settings; (b) applying `Uni-OPD` to broader capability merging scenarios, such as agentic planning, tool use, and long-horizon tasks; and (c) uncovering the mechanistic principles of OPD, particularly how it shapes training dynamics and parameter geometry.

AI USE STATEMENT

In this work, we used generative AI tools for editing the manuscript to improve readability. We have not used generative AI tools for designing the research methodology or experiments, implementing methods, interpreting results, or developing theoretical frameworks; these tasks were carried out entirely by the authors. We have reviewed all AI-assisted edits. We take responsibility for the final content of this work, including all text, claims, and artifacts.

ETHICS STATEMENT

Our work studies on-policy distillation using publicly available models and benchmarks, and does not involve human subjects or the release of new personally identifiable data. Accordingly, we do not identify any major ethical or societal concerns arising from this work.

REPRODUCIBILITY STATEMENT

To facilitate a clear understanding of our contributions and support broader adoption of our work, we provide extensive materials. In the main text, we detail the key components in §3 and report the main experimental results in §4. In the appendix, we further elaborate on Data Balancing Details (§A), Margin Calibration Details (§B), Training Details (§C), and Evaluation Details (§D), which together should be sufficient to reproduce our results. We provide code and example scripts in our anonymous repository, and will release the full training data and model checkpoints to accelerate future research.

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 preprint arXiv:2303.08774*, 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 *Proceedings of the Twelfth International Conference on Learning Representations*, 2024.

AI-MO. AIME 2024. https://huggingface.co/datasets/AI-MO/aimo-validation-aime, 2024.

AI@Meta. Introducing Llama 3.1: Our most capable models to date. https://ai.meta.com/blog/meta-llama-3-1, 2024a.

AI@Meta. Llama 3 model card. https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md, 2024b.

Chenxin An, Zhihui Xie, Xiaonan Li, Lei Li, Jun Zhang, Shansan Gong, Ming Zhong, Jingjing Xu, Xipeng Qiu, Mingxuan Wang, and Lingpeng Kong. POLARIS: A post-training recipe for scaling reinforcement learning on advanced reasoning models. https://hkunlp.github.io/blog/2025/Polaris, 2025.

Anthropic. Model card and evaluations for Claude models. https://www-cdn.anthropic.com/5c49cc247484cecf107c699baf29250302e5da70/claude-2-model-card.pdf, 2023a.

Anthropic. Introducing Claude. https://www.anthropic.com/index/introducing-claude, 2023b.

Anthropic. The Claude 3 model family: Opus, Sonnet, Haiku. https://www-cdn.anthropic.com/de8ba9b01c9ab7cbabf5c33b80b7bbc618857627/Model_Card_Claude_3.pdf, 2024.

Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-VL: A versatile vision-language model for understanding, localization, text reading, and beyond. *arXiv preprint arXiv:2308.12966*, 2023.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-VL Technical Report. *arXiv preprint arXiv:2511.21631*, 2025a.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2.5-VL Technical Report. *arXiv preprint arXiv:2502.13923*, 2025b.

Mislav Balunović, Jasper Dekoninck, Ivo Petrov, Nikola Jovanović, and Martin Vechev. MathArena: Evaluating LLMs on uncontaminated math competitions. *arXiv preprint arXiv:2505.23281*, 2025.

Hritik Bansal, Devendra Singh Sachan, Kai-Wei Chang, Aditya Grover, Gargi Ghosh, Wen-tau Yih, and Ramakanth Pasunuru. HoneyBee: Data recipes for vision-language reasoners. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 26262–26273, 2026.

Edward Beeching, Clémentine Fourrier, Nathan Habib, Sheon Han, Nathan Lambert, Nazneen Rajani, Omar Sanseviero, Lewis Tunstall, and Thomas Wolf. Open LLM Leaderboard. https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard, 2023.

Walid Bousselham, Hilde Kuehne, and Cordelia Schmid. VOLD: Reasoning transfer from LLMs to vision-language models via on-policy distillation. *arXiv preprint arXiv:2510.23497*, 2025.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *Proceedings of Advances in Neural Information Processing Systems*, 2020.

Di Cao, Dongjie Fu, Hai Yu, Siqi Zheng, Xu Tan, and Tao Jin. X-OPD: Cross-modal on-policy distillation for capability alignment in speech LLMs. *arXiv preprint arXiv:2603.24596*, 2026.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? Try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Ganqu Cui, Lifan Yuan, Zefan Wang, Hanbin Wang, Yuchen Zhang, Jiacheng Chen, Wendi Li, Bingxiang He, Yuchen Fan, Tianyu Yu, et al. Process reinforcement through implicit rewards. *arXiv preprint arXiv:2502.01456*, 2025.

Wenliang Dai, Junnan Li, Dongxu Li, Anthony Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale N Fung, and Steven Hoi. InstructBLIP: Towards general-purpose vision-language models with instruction tuning. In *Proceedings of Advances in Neural Information Processing Systems*, 2023.

Yuqian Fu, Haohuan Huang, Kaiwen Jiang, Jiacai Liu, Zhuo Jiang, Yuanheng Zhu, and Dongbin Zhao. Revisiting On-Policy Distillation: Empirical failure modes and simple fixes. *arXiv preprint arXiv:2603.25562*, 2026.

Chang Gao, Chujie Zheng, Xiong-Hui Chen, Kai Dang, Shixuan Liu, Bowen Yu, An Yang, Shuai Bai, Jingren Zhou, and Junyang Lin. Soft Adaptive Policy Optimization. *arXiv preprint arXiv:2511.20347*, 2025.

Naibin Gu, Chenxu Yang, Qingyi Si, Chuanyu Qin, Dingyu Yao, Peng Fu, Zheng Lin, Weiping Wang, Nan Duan, and Jiaqi Wang. Co-evolving policy distillation. *arXiv preprint arXiv:2604.27083*, 2026.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. MiniLLM: On-policy distillation of large language models. *arXiv preprint arXiv:2306.08543*, 2023.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025a.

Yiju Guo, Wenkai Yang, Zexu Sun, Ning Ding, Zhiyuan Liu, and Yankai Lin. Learning to Focus: Causal attention distillation via gradient-guided token pruning. *arXiv preprint arXiv:2506.07851*, 2025b.

Changyi He, Yifu Ding, Jinyang Guo, Ruihao Gong, Haotong Qin, and Xianglong Liu. DA-KD: difficulty-aware knowledge distillation for efficient large language models. In *Proceedings of the Forty-second International Conference on Machine Learning*, 2025a.

Qiangqiang He, Jin Li, and MingCai Chen. Calibrating teacher–student discrepancy for on-policy distillation. *arXiv preprint arXiv:2609.21619*, 2026.

Zhiwei He, Tian Liang, Jiahao Xu, Qiuzhi Liu, Xingyu Chen, Yue Wang, Linfeng Song, Dian Yu, Zhenwen Liang, Wenxuan Wang, et al. DeepMath-103K: A large-scale, challenging, decontaminated, and verifiable mathematical dataset for advancing reasoning. *arXiv preprint arXiv:2504.11456*, 2025b.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *Proceedings of the International Conference on Learning Representations*, 2021.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Wenjin Hou, Wei Liu, Han Hu, Xiaoxiao Sun, Serena Yeung-Levy, and Hehe Fan. Seeing is believing? a benchmark for multimodal large language models on visual illusions and anomalies. *arXiv preprint arXiv:2602.01816*, 2026.

Jonas Hübotter, Frederike Lübeck, Lejs Behric, Anton Baumann, Marco Bagatella, Daniel Marta, Ido Hakimi, Idan Shenfeld, Thomas Kleine Buening, Carlos Guestrin, et al. Reinforcement learning via self-distillation. *arXiv preprint arXiv:2601.20802*, 2026.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. GPT-4o System Card. *arXiv preprint arXiv:2410.21276*, 2024.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. LiveCodeBench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint arXiv:2403.07974*, 2024.

Ijun Jang, Jewon Yeom, Juan Yeo, Hyunggu Lim, and Taesup Kim. Stable on-policy distillation through adaptive target reformulation. *arXiv preprint arXiv:2601.07155*, 2026.

Woogyeol Jin, Taywon Min, Yongjin Yang, Swanand Ravindra Kadhe, Yi Zhou, Dennis Wei, Nathalie Baracaldo, and Kimin Lee. Entropy-aware on-policy distillation of language models. *arXiv preprint arXiv:2603.07079*, 2026.

Simran Kaur, Narutatsu Ri, Yinghui He, Liam Fowl, and Sanjeev Arora. Rethinking on-policy self-distillation for thinking models. *arXiv preprint arXiv:2607.05184*, 2026.

Aniruddha Kembhavi, Mike Salvato, Eric Kolve, Minjoon Seo, Hannaneh Hajishirzi, and Ali Farhadi. A diagram is worth a dozen images. In *Proceedings of the European Conference on Computer Vision*, pp. 235–251, 2016.

Minsang Kim and Seung Jun Baek. Explain in your own words: Improving reasoning via token-selective dual knowledge distillation. *arXiv preprint arXiv:2603.13260*, 2026.

Jongwoo Ko, Tianyi Chen, Sungnyun Kim, Tianyu Ding, Luming Liang, Ilya Zharkov, and Se-Young Yun. DistiLLM-2: A contrastive approach boosts the distillation of LLMs. *arXiv preprint arXiv:2503.07067*, 2025.

Jongwoo Ko, Sara Abdali, Young Jin Kim, Tianyi Chen, and Pashmina Cameron. Scaling reasoning efficiently via relaxed on-policy distillation. *arXiv preprint arXiv:2603.11137*, 2026.

Kalle Kujanpää, Pekka Marttinen, Harri Valpola, and Alexander Ilin. Efficient knowledge injection in LLMs via self-distillation. *arXiv preprint arXiv:2412.14964*, 2024.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

Xin Lai, Zhuotao Tian, Yukang Chen, Yanwei Li, Yuhui Yuan, Shu Liu, and Jiaya Jia. LISA: Reasoning segmentation via large language model. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2024.

Gengluo Li, Xingyu Wan, Shangpin Peng, Weinong Wang, Hao Feng, Yongkun Du, Binghong Wu, Zheng Ruan, Zhiqiong Lu, Liang Wu, Pengyuan Lyu, Huawen Shen, Zibin Lin, Shijing Hu, Jieneng Yang, Hongbing Wen, Guanghua Yu, Hong Liu, Bochao Wang, Can Ma, Han Hu, Chengquan Zhang, and Yu Zhou. HunyuanOCR-1.5: Making lightweight OCR VLMs faster and better. *arXiv preprint arXiv:2607.04884*, 2026a.

Jiaze Li, Hao Yin, Haoran Xu, Boshen Xu, Wenhui Tan, Zewen He, Jianzhong Ju, Zhenbo Luo, and Jian Luan. Video-OPD: Efficient post-training of multimodal large language models for temporal video grounding via on-policy distillation. *arXiv preprint arXiv:2602.02994*, 2026b.

Jingyao Li, Senqiao Yang, Sitong Wu, Han Shi, Chuanyang Zheng, Hong Xu, and Jiaya Jia. Logits-based finetuning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, 2025.

Yaxuan Li, Yuxin Zuo, Bingxiang He, Jinqian Zhang, Chaojun Xiao, Cheng Qian, Tianyu Yu, Huanang Gao, Wenkai Yang, Zhiyuan Liu, et al. Rethinking on-policy distillation of large language models: Phenomenology, mechanism, and recipe. *arXiv preprint arXiv:2604.13016*, 2026c.

Stephanie Lin, Jacob Hilton, and Owain Evans. TruthfulQA: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, 2022.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. DeepSeek-V3 Technical Report. *arXiv preprint arXiv:2412.19437*, 2024a.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *Proceedings of Advances in Neural Information Processing Systems*, 2023a.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2024b.

Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. LLaVA-NeXT: Improved reasoning, OCR, and world knowledge. https://llava-vl.github.io/blog/2024-01-30-llava-next, 2024c.

Jiaheng Liu, Chenchen Zhang, Jinyang Guo, Yuanxing Zhang, Haoran Que, Ken Deng, Zhiqi Bai, Jie Liu, Ge Zhang, Jiakai Wang, et al. DDK: Distilling domain knowledge for efficient large language models. In *Proceedings of Advances in Neural Information Processing Systems*, 2024d.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by ChatGPT really correct? rigorous evaluation of large language models for code generation. In *Proceedings of Advances in Neural Information Processing Systems*, volume 36, pp. 21558–21572, 2023b.

Lingyuan Liu and Mengxiang Zhang. Less is More: Selective reflection for compatible and efficient knowledge distillation in large language models. *arXiv preprint arXiv:2508.06135*, 2025.

Yijun Liu, Jiequan Cui, Zhuotao Tian, Senqiao Yang, Qingdong He, Xiaoling Wang, and Jingyong Su. Typicalness-aware learning for failure detection. In *Proceedings of Advances in Neural Information Processing Systems*, 2024e.

Yuqi Liu, Bohao Peng, Zhisheng Zhong, Zihao Yue, Fanbin Lu, Bei Yu, and Jiaya Jia. Seg-Zero: Reasoning-chain guided segmentation via cognitive reinforcement. *arXiv preprint arXiv:2503.06520*, 2025.

Kevin Lu and Thinking Machines Lab. On-policy distillation. *Thinking Machines Lab: Connectionism*, 2025. doi: 10.64434/tml.20251026. https://thinkingmachines.ai/blog/on-policy-distillation.

Feng Luo, Yu-Neng Chuang, Guanchu Wang, Zicheng Xu, Xiaotian Han, Tianyi Zhang, and Vladimir Braverman. Demystifying OPD: Length inflation and stabilization strategies for large language models. *arXiv preprint arXiv:2604.08527*, 2026.

Ahmed Masry, Xuan Long Do, Jia Qing Tan, Shafiq Joty, and Enamul Hoque. ChartQA: A benchmark for question answering about charts with visual and logical reasoning. In *Findings of the Association for Computational Linguistics: ACL*, 2022.

Minesh Mathew, Dimosthenis Karatzas, and CV Jawahar. DocVQA: A dataset for VQA on document images. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*, pp. 2200–2209, 2021.

Minesh Mathew, Viraj Bagal, Rubèn Tito, Dimosthenis Karatzas, Ernest Valveny, and CV Jawahar. InfographicVQA. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*, pp. 1697–1706, 2022.

Yu Meng, Mengzhou Xia, and Danqi Chen. SimPO: Simple preference optimization with a reference-free reward. In *Proceedings of Advances in Neural Information Processing Systems*, 2024.

OpenAI. GPT-4V(ision) System Card. https://openai.com/index/gpt-4v-system-card, 2023.

OpenCompass. AIME 2025. https://huggingface.co/datasets/opencompass/AIME2025, 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 *Proceedings of Advances in Neural Information Processing Systems*, volume 35, 2022.

Carlos Miguel Patiño, Kashif Rasul, Quentin Gallouédec, Ben Burtenshaw, Sergio Paniego, Vaibhav Srivastav, Thibaud Frere, Ed Beeching, Lewis Tunstall, Leandro von Werra, and Thomas Wolf. Unlocking on-policy distillation for any model family. https://huggingface.co/spaces/HuggingFaceH4/on-policy-distillation, 2025.

Shangpin Peng, Weinong Wang, Zhuotao Tian, Senqiao Yang, Xing Wu, Haotian Xu, Chengquan Zhang, Takashi Isobe, Baotian Hu, and Min Zhang. Uni-DPO: A unified paradigm for dynamic preference optimization of LLMs. *arXiv preprint arXiv:2506.10054*, 2025a.

Shangpin Peng, Senqiao Yang, Li Jiang, and Zhuotao Tian. Mitigating object hallucinations via sentence-level early intervention. In *Proceedings of the IEEE International Conference on Computer Vision*, 2025b.

Shangpin Peng, Gengluo Li, Xingyu Wan, Chengquan Zhang, Hao Feng, Binghong Wu, Huawen Shen, Weinong Wang, Ziyi Cai, Zhuotao Tian, Han Hu, Can Ma, and Yu Zhou. ChartArena: Benchmarking chart parsing across languages, scenarios, and formats. *arXiv preprint arXiv:2606.01348*, 2026.

Runqi Qiao, Qiuna Tan, Guanting Dong, MinhuiWu MinhuiWu, Chong Sun, Xiaoshuai Song, Jiapeng Wang, Zhuoma Gongque, Shanglin Lei, Yifan Zhang, et al. We-Math: Does your large multimodal model achieve human-like mathematical reasoning? In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*, pp. 20023–20070, 2025.

Libo Qin, Qiguang Chen, Yuhang Zhou, Zhi Chen, Yinghui Li, Lizi Liao, Min Li, Wanxiang Che, and Philip S Yu. Multilingual large language model: A survey of resources, taxonomy and frontiers. *arXiv preprint arXiv:2404.04925*, 2024.

Tianyuan Qu, Longxiang Tang, Bohao Peng, Senqiao Yang, Bei Yu, and Jiaya Jia. Does your vision-language model get lost in the long video sampling dilemma? *arXiv preprint arXiv:2503.12496*, 2025.

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

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *Proceedings of the International Conference on Machine Learning*, 2021.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial Winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

Tong Shao, Zhuotao Tian, Hang Zhao, and Jingyong Su. Explore the potential of CLIP for training-free open vocabulary semantic segmentation. In *Proceedings of the European Conference on Computer Vision*, 2024a.

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 preprint arXiv:2402.03300*, 2024b.

Idan Shenfeld, Mehul Damani, Jonas Hübotter, and Pulkit Agrawal. Self-distillation enables continual learning. *arXiv preprint arXiv:2601.19897*, 2026.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-LM: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*, 2019.

Mingyang Song and Mao Zheng. A survey of on-policy distillation for large language models. *arXiv preprint arXiv:2604.00626*, 2026.

Alex Stein, Furong Huang, and Tom Goldstein. GATES: Self-distillation under privileged context with consensus gating. *arXiv preprint arXiv:2602.20574*, 2026.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1*, 2019.

Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024.

Hunyuan Vision Team, Pengyuan Lyu, Xingyu Wan, Gengluo Li, Shangpin Peng, Weinong Wang, Liang Wu, Huawen Shen, Yu Zhou, Canhui Tang, et al. HunyuanOCR Technical Report. *arXiv preprint arXiv:2511.19575*, 2025.

Kimi Team, Tongtong Bai, Yifan Bai, Yiping Bao, SH Cai, Yuan Cao, Y Charles, HS Che, Cheng Chen, Guanduo Chen, et al. Kimi K2.5: Visual Agentic Intelligence. *arXiv preprint arXiv:2602.02276*, 2026.

Zhuotao Tian, Michelle Shu, Pengyuan Lyu, Ruiyu Li, Chao Zhou, Xiaoyong Shen, and Jiaya Jia. Learning shape-aware embedding for scene text detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2019.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Junjie Wang, Bin Chen, Yulin Li, Bin Kang, Yichi Chen, and Zhuotao Tian. DeCLIP: Decoupled learning for open-vocabulary dense perception. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025.

Ke Wang, Junting Pan, Weikang Shi, Zimu Lu, Houxing Ren, Aojun Zhou, Mingjie Zhan, and Hongsheng Li. Measuring multimodal mathematical reasoning with MATH-Vision dataset. In *Proceedings of Advances in Neural Information Processing Systems*, volume 37, pp. 95095–95169, 2024a.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-VL: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024b.

Yinjie Wang, Xuyang Chen, Xiaolong Jin, Mengdi Wang, and Ling Yang. OpenClaw-RL: Train any agent simply by talking. *arXiv preprint arXiv:2603.10165*, 2026.

Xumeng Wen, Zihan Liu, Shun Zheng, Zhijian Xu, Shengyu Ye, Zhirong Wu, Xiao Liang, Yang Wang, Junjie Li, Ziming Miao, et al. Reinforcement learning with verifiable rewards implicitly incentivizes correct reasoning in base LLMs. *arXiv preprint arXiv:2506.14245*, 2025.

Yecheng Wu, Song Han, and Hai Cai. Lightning OPD: Efficient post-training for large reasoning models with offline on-policy distillation. *arXiv preprint arXiv:2604.13010*, 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 preprint arXiv:2601.02780*, 2026.

Yijia Xiao, Edward Sun, Tianyu Liu, and Wei Wang. LogicVista: Multimodal LLM logical reasoning benchmark in visual contexts. *arXiv preprint arXiv:2407.04973*, 2024.

Yan Xie, Sijie Zhu, Tiansheng Wen, Bo Chen, and Yifei Wang. On the position bias of on-policy distillation. *arXiv preprint arXiv:2606.22600*, 2026.

Jing Xiong, Hui Shen, Shansan Gong, Yuxin Cheng, Jianghan Shen, Chaofan Tao, Haochen Tan, Haoli Bai, Lifeng Shang, and Ngai Wong. OVD: On-policy verbal distillation. *arXiv preprint arXiv:2601.21968*, 2026.

Anyi Xu, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, Chenchen Ling, et al. DeepSeek-V4: Towards highly efficient million-token context intelligence. *arXiv preprint arXiv:2606.19348*, 2026a.

Weiye Xu, Jiahao Wang, Weiyun Wang, Zhe Chen, Wengang Zhou, Aijun Yang, Lewei Lu, Houqiang Li, Xiaohua Wang, Xizhou Zhu, et al. VisuLogic: A benchmark for evaluating visual reasoning in multi-modal large language models. *arXiv preprint arXiv:2504.15279*, 2025.

Xiaohan Xu, Ming Li, Chongyang Tao, Tao Shen, Reynold Cheng, Jinyang Li, Can Xu, Dacheng Tao, and Tianyi Zhou. A survey on knowledge distillation of large language models. *arXiv preprint arXiv:2402.13116*, 2024.

Yuanda Xu, Hejian Sang, Zhengze Zhou, Ran He, and Zhipeng Wang. PACED: Distillation at the frontier of student competence. *arXiv preprint arXiv:2603.11178*, 2026b.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2.5 Technical Report. *arXiv preprint arXiv:2412.15115*, 2024a.

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 preprint arXiv:2505.09388*, 2025a.

Chenxu Yang, Chuanyu Qin, Qingyi Si, Minghui Chen, Naibin Gu, Dingyu Yao, Zheng Lin, Weiping Wang, Jiaqi Wang, and Nan Duan. Self-distilled RLVR. *arXiv preprint arXiv:2604.03128*, 2026a.

Senqiao Yang, Jiaming Liu, Ray Zhang, Mingjie Pan, Zoey Guo, Xiaoqi Li, Zehui Chen, Peng Gao, Yandong Guo, and Shanghang Zhang. LiDAR-LLM: Exploring the potential of large language models for 3d LiDAR understanding. *arXiv preprint arXiv:2312.14074*, 2023a.

Senqiao Yang, Tianyuan Qu, Xin Lai, Zhuotao Tian, Bohao Peng, Shu Liu, and Jiaya Jia. An improved baseline for reasoning segmentation with large language model. *arXiv preprint arXiv:2312.17240*, 2023b.

Senqiao Yang, Zhuotao Tian, Li Jiang, and Jiaya Jia. Unified language-driven zero-shot domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2024b.

Senqiao Yang, Yukang Chen, Zhuotao Tian, Chengyao Wang, Jingyao Li, Bei Yu, and Jiaya Jia. VisionZip: Longer is better but not necessary in vision language models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2025b.

Wenkai Yang, Weijie Liu, Ruobing Xie, Kai Yang, Saiyong Yang, and Yankai Lin. Learning beyond teacher: Generalized on-policy distillation with reward extrapolation. *arXiv preprint arXiv:2602.12125*, 2026b.

Zhuolin Yang, Zihan Liu, Yang Chen, Wenliang Dai, Boxin Wang, Sheng-Chieh Lin, Chankyu Lee, Yangyi Chen, Dongfu Jiang, Jiafan He, et al. Nemotron-Cascade 2: Post-training LLMs with cascade RL and multi-domain on-policy distillation. *arXiv preprint arXiv:2603.19220*, 2026c.

Tianzhu Ye, Li Dong, Zewen Chi, Xun Wu, Shaohan Huang, and Furu Wei. Black-box on-policy distillation of large language models. *arXiv preprint arXiv:2511.10643*, 2025.

Tianzhu Ye, Li Dong, Xun Wu, Shaohan Huang, and Furu Wei. On-policy context distillation for language models. *arXiv preprint arXiv:2602.12275*, 2026.

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. In *Proceedings of Advances in Neural Information Processing Systems*, volume 38, pp. 113222–113244, 2026.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019.

Aohan Zeng, Xin Lv, Zhenyu Hou, Zhengxiao Du, Qinkai Zheng, Bin Chen, Da Yin, Chendi Ge, Chenghua Huang, Chengxing Xie, et al. GLM-5: from vibe coding to agentic engineering. *arXiv preprint arXiv:2602.15763*, 2026.

Dongxu Zhang, Zhichao Yang, Sepehr Janghorbani, Jun Han, Andrew Ressler II, Qian Qian, Gregory D Lyng, Sanjit Singh Batra, and Robert E Tillman. Fast and effective on-policy distillation from reasoning prefixes. *arXiv preprint arXiv:2602.15260*, 2026a.

Kaichen Zhang, Bo Li, Peiyuan Zhang, Fanyi Pu, Joshua Adrian Cahyono, Kairui Hu, Shuai Liu, Yuanhan Zhang, Jingkang Yang, Chunyuan Li, et al. LMMs-Eval: Reality check on the evaluation of large multimodal models. In *Findings of the Association for Computational Linguistics: NAACL*, 2025a.

Kaichen Zhang, Keming Wu, Zuhao Yang, Bo Li, Kairui Hu, Bin Wang, Ziwei Liu, Xingxuan Li, and Lidong Bing. OpenMMReasoner: Pushing the frontiers for multimodal reasoning with an open and general recipe. *arXiv preprint arXiv:2511.16334*, 2025b.

Songming Zhang, Xue Zhang, Tong Zhang, Bojie Hu, Yufeng Chen, and Jinan Xu. KDFlow: A user-friendly and efficient knowledge distillation framework for large language models. *arXiv preprint arXiv:2603.01875*, 2026b.

Yi Zhang, Bolin Ni, Xin-Sheng Chen, Heng-Rui Zhang, Yongming Rao, Houwen Peng, Qinglin Lu, Han Hu, Meng-Hao Guo, and Shi-Min Hu. Bee: A high-quality corpus and full-stack suite to unlock advanced fully open MLLMs. *arXiv preprint arXiv:2510.13795*, 2025c.

Jiaxing Zhao, Xihan Wei, and Liefeng Bo. R1-Omni: Explainable Omni-Multimodal emotion recognition with reinforcement learning. *arXiv preprint arXiv:2503.05379*, 2025.

Shiwan Zhao, Zhihu Wang, Xuyang Zhao, Jiaming Zhou, Caiyue Xu, Chenfei Liu, Liting Zhang, Yuhang Jia, Yanzhe Zhang, Hualong Yu, et al. Large language model post-training: A unified view of off-policy and on-policy learning. *arXiv preprint arXiv:2604.07941*, 2026a.

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 preprint arXiv:2601.18734*, 2026b.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, et al. Group Sequence Policy Optimization. *arXiv preprint arXiv:2507.18071*, 2025a.

Chujie Zheng, Ziqi Wang, Heng Ji, Minlie Huang, and Nanyun Peng. Model extrapolation expedites alignment. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1025–1041, 2025b.

Zhisheng Zhong, Chengyao Wang, Yuqi Liu, Senqiao Yang, Longxiang Tang, Yuechen Zhang, Jingyao Li, Tianyuan Qu, Yanwei Li, Yukang Chen, et al. Lyra: An efficient and speech-centric framework for Omni-cognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3694–3704, 2025.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, et al. LIMA: Less is more for alignment. In *Proceedings of Advances in Neural Information Processing Systems*, volume 36, pp. 55006–55021, 2023a.

Guorui Zhou, Honghui Bao, Jiaming Huang, Jiaxin Deng, Jinghao Zhang, Junda She, Kuo Cai, Lejian Ren, Lu Ren, Qiang Luo, et al. OpenOneRec Technical Report. *arXiv preprint arXiv:2512.24762*, 2025.

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 preprint arXiv:2311.07911*, 2023b.

Deyao Zhu, Xiaoqian Shen, Xiang Li, Mohamed Elhoseiny, et al. MiniGPT-4: Enhancing vision-language understanding with advanced large language models. In *Proceedings of the International Conference on Learning Representations*, 2024.

Chengke Zou, Xingang Guo, Rui Yang, Junyu Zhang, Bin Hu, and Huan Zhang. DynaMath: A dynamic visual benchmark for evaluating mathematical reasoning robustness of vision language models. In *Proceedings of the International Conference on Learning Representations*, pp. 48337–48383, 2025.

APPENDIX OUTLINE

This material provides supplementary details to the main paper, including the following sections:

- (A) **Data Balancing Details**
  - (A.1) Offline Difficulty-Aware Data Balancing
  - (A.2) Online Correctness-Aware Data Balancing
- (B) **Margin Calibration Details**
  - (B.1) Failure Modes of Teacher Supervision
  - (B.2) Order Consistency of Trajectory-level Returns
  - (B.3) Order-Consistency Violations and Their Harm
  - (B.4) Statistical Evidence for Order-Consistency Violations
  - (B.5) Outcome-Guided Margin Calibration
  - (B.6) Qualitative Visualization of Margin Calibration
- (C) **Training Details**
  - (C.1) Training Setup
  - (C.2) Training Pseudocode
  - (C.3) Training Data
  - (C.4) Training Reward Acquisition
  - (C.5) Training Dynamics
  - (C.6) Training Complexity
- (D) **Evaluation Details**
  - (D.1) Evaluation Benchmarks
  - (D.2) Baselines
  - (D.3) Evaluation Setup
- (E) **Further Evaluations**
  - (E.1) More Evaluation Results
  - (E.2) Downstream Task Evaluation
  - (E.3) Further Ablation
  - (E.4) Analysis and Takeaways
- (F) **Related Work**
  - (F.1) Multimodal Large Language Models
  - (F.2) Reinforcement Learning
  - (F.3) On-Policy Distillation
- (G) **Case Studies**

A Data Balancing Details

In this section, we provide a detailed exposition of data balancing strategies in our proposed `Uni-OPD` framework, including formulations and implementations. These strategies operate on complementary levels: offline difficulty-aware balancing (§A.1) shapes the *prompt-level* difficulty distribution before training, while online correctness-aware balancing (§A.2) regulates the *intra-group* composition of correct and incorrect trajectories during rollout.

A.1 Offline Difficulty-Aware Data Balancing

Here, we provide a detailed description of our offline difficulty-aware data balancing strategy.

Offline rollout sampling. Before training, we perform a one-time offline rollout pass over the entire training dataset using the student model (*e.g.,* Qwen3-4B). For each training prompt, the student is prompted to generate $N = 8$ independent candidate responses, which serve as the basis for subsequent difficulty estimation.

The rollouts are produced with vLLM (Kwon et al., 2023) under the same prompt template that will later be used at training time, so that the estimated difficulty reflects the actual input format the student will see. The decoding configuration is kept fixed throughout this offline phase: we use temperature $= 1.0$, top-$p = 0.95$, top-$k = 50$, and a maximum response length of $16{,}384$ tokens. For each instance, we then verify the correctness of its N candidate responses with the task-specific verifier (described in §C.4) and record the number of correct ones. The resulting empirical pass rate k/N serves as our proxy for the instance's difficulty: a lower pass rate indicates a harder example, while a higher pass rate indicates an easier one.

Limitations of aggressive difficulty filtering. Prior work on RL optimization, such as GRPO, often relies on a heuristic pre-training filter that simply discards "trivial" samples such as all-correct cases, because these instances yield zero advantage and therefore provide essentially no learning signal. POLARIS (An et al., 2025), for example, reports that removing the easiest samples leads to consistent performance gains, and argues that keeping an unfiltered dataset can hinder training.

In the token-level reward OPD setting, however, we find that such aggressive filtering is counter-productive. Empirically, removing any specific difficulty tier, whether the easiest or the hardest, consistently hurts final performance. A plausible explanation is that each tier contributes a distinct pattern of token-level credit: easy instances calibrate the student's baseline behavior, intermediate instances provide the richest contrastive signals between correct and incorrect trajectories, and hard instances expose the student to diverse, non-trivial solution paths. Therefore, dropping any tier both distorts the overall distribution of token-level credit and narrows the space of solution patterns to which the student is exposed.

Difficulty-aware data balancing. Motivated by this observation, we adopt a difficulty-aware balancing scheme that deliberately preserves the full spectrum of difficulty while reweighting its different regions, rather than truncating them. Concretely, after the offline rollout pass, we examine the empirical distribution over the number of correct responses out of N. Across our training sources, we observe two recurring shapes: (i) a **U-shaped** distribution, where both very easy and very hard instances dominate while intermediate ones are sparse; and (ii) a **mirrored-J-shaped** distribution, where hard instances dominate and the mass decays toward the easy end.

We treat the two shapes slightly differently. For U-shaped distributions, we upsample instances of intermediate difficulty, namely those with 1–7 correct responses out of $N = 8$, so as to fill in the under-represented middle region. For mirrored-J-shaped distributions, we instead upsample every instance with at least one correct response, *i.e.,* 1–8 correct out of $N = 8$, to counteract the mass concentrated at the hard end. In both cases, the effect of the reweighting is to flatten the overall difficulty distribution and to ensure that the token-level credit signals arriving during training are more evenly spread across difficulty levels. Empirically, as shown in Fig. 3, we find that this rebalancing consistently leads to better final performance than either no filtering or the conventional drop-the-easy-cases strategy.

A.2 Online Correctness-Aware Data Balancing

In this section, we detail the online correctness-aware data balancing strategy that operates during rollout. While the offline difficulty-aware balancing in §A.1 controls the *prompt-level* difficulty distribution before training, the composition of correct and incorrect trajectories *within a rollout group* still varies dramatically as the student evolves. This subsection describes how we regulate such intra-group composition online.

Motivation. For each prompt $\boldsymbol{q}$ we sample N on-policy trajectories $\{\boldsymbol{\tau}_i\}_{i=1}^{N}$ and split them into a positive set $S_+(\boldsymbol{q})$ and a negative set $S_-(\boldsymbol{q})$ based on the outcome reward R_i, as discussed in §3.4. As training proceeds, many prompts exhibit degenerate outcome distributions: either $|S_-(\boldsymbol{q})| \ll N$ (the student nearly masters $\boldsymbol{q}$) or $|S_+(\boldsymbol{q})| \ll N$ (the student often fails on $\boldsymbol{q}$). In both cases, the outcome-level contrast vanishes and the outcome-guided margin calibration may not provide corrective signal, since the prompt-level margin $m(\boldsymbol{q})$ is undefined. If left unregulated, such degenerate groups dominate the batch and drive the student into local optima with shrinking exploration.

Online correctness-aware balancing. To preserve sufficient outcome diversity throughout training, we maintain a target correct-to-total ratio $\gamma^\star \in (0,1)$ within each rollout group. We use $\gamma^\star \approx 0.5$ by default, *i.e.,* a 1:1 correct-to-incorrect ratio, so positive and negative trajectories are balanced. At each training step, for every freshly rolled-out group $\{\boldsymbol{\tau}_i\}_{i=1}^{N}$ of a prompt $\boldsymbol{q}$, we let $\gamma(\boldsymbol{q}) = \sum_{i=1}^{N} \mathbf{1}\{R_i = 1\}/N$ denote the current correct-to-total ratio inside this group. Whenever $|\gamma(\boldsymbol{q}) - \gamma^\star| > \epsilon$ for a tolerance ϵ, we filter out trajectories from the over-represented side (correct or incorrect) of the group, so that the ratio over the retained trajectories is pulled back to the $\gamma^\star \pm \epsilon$ interval. Subsampling is performed inside the group, which keeps the intra-group difficulty distribution intact and avoids biasing the prompt-level difficulty spectrum inherited from offline balancing.

B Margin Calibration Details

In this section, we provide a detailed exposition of the outcome-guided margin calibration in `Uni-OPD`. We begin with the typical ways in which teacher supervision fails on student rollouts (§B.1). We then recall the order-consistency criterion for trajectory-level returns (§B.2), and formalize what it means for teacher supervision to violate this criterion and analyze the resulting harm (§B.3). We next present a statistical study showing that such violations are prevalent in practice (§B.4), which motivates the two calibration strategies described in §B.5. We finally visualize the two failure modes and the effect of calibration at the token level (§B.6).

B.1 Failure Modes of Teacher Supervision

As shown in Eq. (3), OPD relies on the teacher to provide fine-grained supervision for student-generated trajectories. For effective optimization, this signal should align with overall trajectory correctness. In practice, this alignment is not guaranteed and can fail in several typical ways: **(a) OOD degradation**: when student rollouts enter sparse or out-of-distribution regions relative to the teacher, $\log \pi_{\mathrm{T}}(o_t \mid \cdot)$ may become noisy, disrupting the ranking between correct and incorrect trajectories. **(b) Overestimation of incorrect trajectories**: incorrect trajectories may receive abnormally high scores when their local token patterns align with the high-confidence regions of the teacher. **(c) Underestimation of correct trajectories**: correct trajectories may receive abnormally low scores when their generation paths deviate from the dominant regions of the teacher, thereby suppressing useful reasoning paths. These phenomena suggest that teacher supervision is not always reliable, motivating us to introduce a global anchor for calibrating trajectory-level supervision.

B.2 Order Consistency of Trajectory-Level Returns

This section provides a brief explanation for the order-consistency conditions in Eqs. (7) and (8) of the main paper. The key observation is two-fold. First, treating the entire reasoning rollout as a single macro-action gives $G_{\mathrm{RL}}(\boldsymbol{q}, \boldsymbol{\tau}) = R(\boldsymbol{q}, \boldsymbol{\tau})$, so G_{RL} respects the outcome-induced ordering by construction. Second, under the distillation premise, the trajectory-level distillation return $G_{\mathrm{OPD}}(\boldsymbol{q}, \boldsymbol{\tau})$ is expected to preserve the same ordering, although this is a desideratum rather than a definitional consequence.

Trajectory-as-one-action view of outcome-based RL. In outcome-based RL for reasoning, supervision is provided only at the trajectory level: a rollout $\boldsymbol{\tau}$ receives a single scalar reward $R(\boldsymbol{q}, \boldsymbol{\tau})$ determined by the final answer. Under this view, the trajectory-level return reduces to the outcome reward itself, *i.e.,*

$$G_{\mathrm{RL}}(\boldsymbol{q}, \boldsymbol{\tau}) = R(\boldsymbol{q}, \boldsymbol{\tau})\,. \tag{13}$$

Order consistency under binary rewards. For the binary outcome reward adopted in this work, any $\boldsymbol{\tau}_+ \in S_+(\boldsymbol{q})$ satisfies $R(\boldsymbol{q}, \boldsymbol{\tau}_+) = 1$, while any $\boldsymbol{\tau}_- \in S_-(\boldsymbol{q})$ satisfies $R(\boldsymbol{q}, \boldsymbol{\tau}_-) = 0$. Combined with Eq. (13), we have

$$G_{\mathrm{RL}}(\boldsymbol{q}, \boldsymbol{\tau}_+) = 1 \;\geq\; 0 = G_{\mathrm{RL}}(\boldsymbol{q}, \boldsymbol{\tau}_-)\,, \tag{14}$$

for all $\boldsymbol{\tau}_+ \in S_+(\boldsymbol{q})$ and $\boldsymbol{\tau}_- \in S_-(\boldsymbol{q})$, which recovers Eq. (7) directly.

Extension to soft outcome rewards. The same argument extends to soft outcome rewards, where $R(\boldsymbol{q}, \boldsymbol{\tau}) \in [0, 1]$ (or any bounded interval) measures a graded notion of correctness, *e.g.,* partial credit or a verifier's confidence score. As long as the trajectory partition is defined by thresholding the outcome reward, *i.e.,* $S_+(\boldsymbol{q}) = \{\boldsymbol{\tau} \mid R(\boldsymbol{q}, \boldsymbol{\tau}) \geq \eta\}$ and $S_-(\boldsymbol{q}) = \{\boldsymbol{\tau} \mid R(\boldsymbol{q}, \boldsymbol{\tau}) < \eta\}$ for some threshold η, then by Eq. (13) every positive trajectory attains a return no smaller than that of any negative trajectory, and Eq. (7) still holds. In particular, the binary case is recovered as the special instance $\eta = 1$, $R \in \{0, 1\}$.

From RL return to distillation return. The distillation return $G_{\mathrm{OPD}}(\boldsymbol{q}, \boldsymbol{\tau})$ defined in Eq. (5) plays the same role for OPD training as G_{RL} does for outcome-based RL: it is the trajectory-level aggregation of the token-level supervision that drives the update along the rollout. The distillation premise in §3.4 posits that, relative to the student, the teacher assigns a higher log-likelihood to correct trajectories than incorrect ones. In other words, the teacher's trajectory-level preference is expected to be aligned with the outcome reward, so that G_{OPD} should inherit the same outcome-level ordering as G_{RL}, leading to Eq. (8). Unlike the RL return, however, G_{OPD} is derived from the teacher–student log-probability gap rather than the outcome reward itself, so the ordering is a desired property rather than a guaranteed one. The order-consistency condition in Eq. (8) provides a principled target, and subsequent margin mask and margin shift strategies (§B.5) are designed to enforce it whenever the teacher's supervision violates this property in practice.

B.3 Order-Consistency Violations and Their Harm

The order-consistency criterion in Eq. (8) states a desideratum: within each prompt, the trajectory-level distillation return should rank every correct trajectory above every incorrect one. In this section, we first formalize what it means for teacher supervision to violate this criterion. We then explain why such a violation harms training by analyzing how the trajectory-level return steers the OPD update. Throughout, we consider a non-degenerate prompt $\boldsymbol{q}$, *i.e.,* one for which both $S_+(\boldsymbol{q})$ and $S_-(\boldsymbol{q})$ are non-empty.

Order-consistency violation. We say that the teacher's trajectory-level supervision *violates order consistency* on prompt $\boldsymbol{q}$ if the ordering condition in Eq. (8) fails, *i.e.,* there exists at least one inverted positive–negative pair:

$$\exists\, \boldsymbol{\tau}_+ \in S_+(\boldsymbol{q}),\ \boldsymbol{\tau}_- \in S_-(\boldsymbol{q}) \quad \text{s.t.} \quad G_{\mathrm{OPD}}(\boldsymbol{q}, \boldsymbol{\tau}_+) < G_{\mathrm{OPD}}(\boldsymbol{q}, \boldsymbol{\tau}_-)\,. \tag{15}$$

By the definition of the prompt-level margin in Eq. (9), the worst inverted pair is exactly the one attaining $\min_{\boldsymbol{\tau} \in S_+} G_{\mathrm{OPD}}$ and $\max_{\boldsymbol{\tau} \in S_-} G_{\mathrm{OPD}}$. Hence, a violation occurs on $\boldsymbol{q}$ if and only if

$$m(\boldsymbol{q}) = \min_{\boldsymbol{\tau} \in S_+(\boldsymbol{q})} G_{\mathrm{OPD}}(\boldsymbol{q}, \boldsymbol{\tau}) - \max_{\boldsymbol{\tau} \in S_-(\boldsymbol{q})} G_{\mathrm{OPD}}(\boldsymbol{q}, \boldsymbol{\tau}) < 0\,, \tag{16}$$

which is the operational condition we use throughout. The margin $m(\boldsymbol{q})$ therefore serves as a certificate: $m(\boldsymbol{q}) \geq 0$ guarantees that no inverted pair exists, whereas $m(\boldsymbol{q}) < 0$ signals that at least one violation exists.

Consequence: violations reinforce incorrect trajectories. Recall from §3.4 that $G_{\mathrm{OPD}}(\boldsymbol{q}, \boldsymbol{\tau})$ is the teacher's average log-likelihood preference over the student along $\boldsymbol{\tau}$: a larger value encourages the student to move toward $\boldsymbol{\tau}$, while a smaller one discourages it. When order consistency holds, every correct trajectory carries a larger G_{OPD} than every incorrect one. The trajectories the student

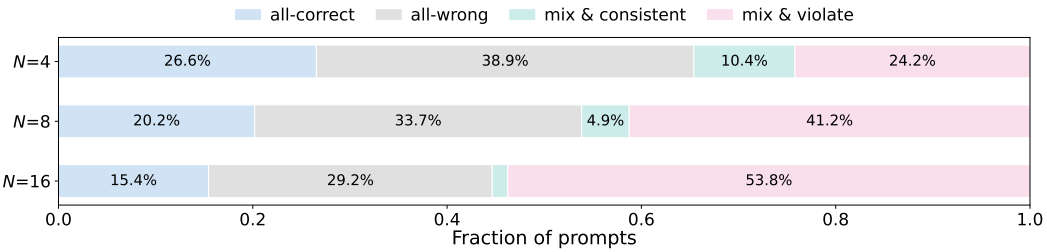


Figure B.1: **Prevalence of order-consistency violations across rollout sizes.** For each prompt, we sample N on-policy trajectories from the student on the math training data and classify them into one of four groups. *All-correct* and *all-wrong* are degenerate prompts with no comparable positive–negative pair; *mix & consistent* denotes non-degenerate prompts satisfying $m(\boldsymbol{q}) \geq 0$; *mix & violate* denotes non-degenerate prompts with $m(\boldsymbol{q}) < 0$. The violating fraction already reaches 24.2% at $N = 4$ and grows rapidly with N, exceeding the consistent fraction.

is encouraged to move toward are therefore exactly the outcome-correct ones, in agreement with the outcome-induced ordering in Eq. (7). A violation reverses this relationship. When $m(\boldsymbol{q}) < 0$, there exists a correct trajectory $\boldsymbol{\tau}_+$ whose return is even smaller than that of some incorrect trajectory $\boldsymbol{\tau}_-$, so the update encourages the student toward the incorrect $\boldsymbol{\tau}_-$ more strongly than toward the correct $\boldsymbol{\tau}_+$. In other words, a violation indicates that the teacher supervision is unreliable in the sense of §B.1. Following such supervision biases the update toward incorrect trajectories and hinders the student's optimization.

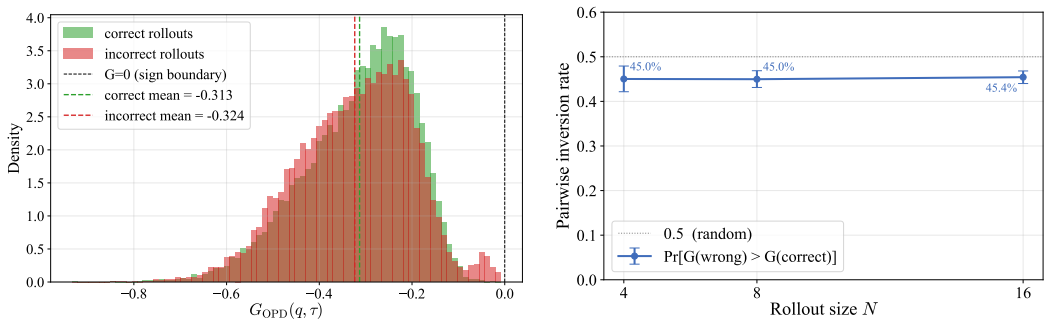
B.4 Statistical Evidence for Order-Consistency Violations

Having formalized order-consistency violations in §B.3, we now provide a statistical study demonstrating that they are prevalent rather than incidental. Unless otherwise stated, we use Qwen3-4B as the student and Qwen3-30B-A3B-Instruct-2507 as the teacher, and sample on-policy trajectories on the textual math training data.

Violations are prevalent and grow with rollout size. We first count how many prompts fail the condition $m(\boldsymbol{q}) \geq 0$ at the prompt level. For each prompt we draw N on-policy trajectories. Following §B.3, we then partition the prompts into four groups: two degenerate ones, *all-correct* and *all-wrong*, for which no comparable positive–negative pair exists and order consistency is vacuous; *mix & consistent*, the non-degenerate prompts for which the lowest-scoring correct trajectory still outranks the highest-scoring incorrect one (*i.e.,* $m(\boldsymbol{q}) \geq 0$); and *mix & violate*, the non-degenerate prompts with $m(\boldsymbol{q}) < 0$. As shown in Fig. B.1, at $N = 4$, 24.2% of prompts violate the condition, substantially more than the 10.4% that remain consistent. The violating fraction increases sharply as N grows, since more sampled trajectories provide more chances to expose an inverted pair, reaching roughly half of all prompts at $N = 16$. This confirms that failing to guarantee $m(\boldsymbol{q}) \geq 0$ is a widespread phenomenon across prompts.

The teacher is globally selective yet biased by degenerate patterns. We next examine the trajectory-level returns aggregated over the entire rollout set, shown in Fig. B.2a. On average, correct rollouts attain a slightly higher G_{OPD} than incorrect ones, indicating that the teacher does possess a certain degree of discriminative and selective ability at the global level. However, the two distributions overlap heavily, and we observe an anomalous concentration of incorrect rollouts near $G_{\mathrm{OPD}} = 0$. Inspecting these cases reveals that they are predominantly degenerate generations that fail to terminate under high-temperature decoding and collapse into repetitive or otherwise low-information continuations (Luo et al., 2026). On the repetitive spans of such trajectories, the teacher tends to assign disproportionately high token-level rewards, which inflates their trajectory-level return toward zero. Such locally favored trajectories are precisely the ones that can outrank correct trajectories and cause the violations analyzed in §B.3. Accumulated over training, they drive up the response length and repetition rate of the student, one of the very failure modes our method is designed to prevent.

Per-prompt orderings are frequently inverted. While the teacher is selective in aggregate, the ordering between correct and incorrect trajectories *within an individual prompt* is far less reliable,



(a) Distribution of G_{OPD} by rollout correctness. (b) Pairwise inversion rate vs. rollout size N.

Figure B.2: **Teacher supervision is globally selective but locally unreliable.** (a) Aggregated over all rollouts, correct trajectories attain a slightly higher mean G_{OPD} than incorrect ones, yet the two distributions overlap heavily; the anomalous mass of incorrect rollouts near $G_{\mathrm{OPD}} = 0$ corresponds to degenerate repetitive generations. (b) Among all comparable correct–incorrect pairs, the fraction in which an incorrect trajectory outranks a correct one stays around 45% and is insensitive to N, indicating that the per-prompt ordering in Eq. (8) is frequently violated.

which is the quantity that order consistency concerns. To measure this directly, we consider all comparable pairs, *i.e.,* every pair consisting of one correct and one incorrect trajectory under the same prompt, and compute the pairwise inversion rate $\Pr[\,G_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau}_-) > G_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau}_+)\,]$. As shown in Fig. B.2b, this rate remains around 45% across all rollout sizes and is largely insensitive to N, so only about 55% of comparable pairs satisfy the ordering required by Eq. (8). In other words, a large fraction of positive–negative pairs fail to meet Eq. (8), confirming that the teacher's local supervision is unreliable at the per-prompt level. This is the problem our approach sets out to solve, and we now describe the two outcome-guided margin calibration strategies for this purpose.

B.5 Outcome-Guided Margin Calibration

In this section, we describe the details of the two outcome-guided margin calibration strategies introduced in §3.4: *Margin Mask* and *Margin Shift*. Both strategies operate on the trajectory-level distillation returns $\{G_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau}_i)\}_{i=1}^{N}$ within a rollout group of a prompt $\boldsymbol{q}$, with the common goal of enforcing the order-consistency condition $m(\boldsymbol{q}) \geq \delta$ (Eq. (10)). They differ in how they repair violations: *Margin Mask* removes the most adversarial trajectories until the condition holds, whereas *Margin Shift* applies a minimal additive correction to restore the margin in closed form.

Margin choices: MinMax vs. Mean. Following the prompt-level margin in Eq. (9), we define the margin between $S_+(\boldsymbol{q})$ and $S_-(\boldsymbol{q})$ in two modes: the MinMax mode uses $\min_{\boldsymbol{\tau}\in S_+} G_{\mathrm{OPD}} - \max_{\boldsymbol{\tau}\in S_-} G_{\mathrm{OPD}}$ and characterizes the worst-case ordering violation; the Mean mode uses $\mathrm{mean}_{\boldsymbol{\tau}\in S_+} G_{\mathrm{OPD}} - \mathrm{mean}_{\boldsymbol{\tau}\in S_-} G_{\mathrm{OPD}}$ and reflects the average-case ordering tendency. MinMax is more conservative (it forces every positive to outrank every negative), while Mean is more lenient and less sensitive to individual outliers. The two modes play different roles in our framework. The MinMax margin of Eq. (9) is the strict diagnostic: it certifies whether any positive–negative pair is inverted, and it is the criterion behind the violation statistics in §B.4. The Mean margin is the default target of the correction itself, so that the magnitude of the shift is decided by the two group means rather than by a single extreme trajectory. The main text instantiates the MinMax case for clarity, and Table E.5 reports both.

Detailed implementation of margin mask. The margin mask strategy discards unreliable trajectories until the prompt-level margin is restored. We implement its fine-grained, data-efficient variant as *Greedy Margin Mask*, which removes the single most adversarial trajectory in each iteration rather than discarding the entire group. Specifically, given the rollout group $\{\boldsymbol{\tau}_i\}_{i=1}^{N}$ of prompt $\boldsymbol{q}$ with trajectory-level returns $\{G_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau}_i)\}_{i=1}^{N}$, we sort the positives in ascending order of G_{OPD} (so the worst correct trajectory comes first) and the negatives in descending order (so the best incorrect trajectory comes first). At each iteration, we compute the margin improvement obtained by removing the front of each sorted list and greedily dropping the side that yields the larger improvement. The iteration terminates once (i) the target margin $m(\boldsymbol{q}) \geq \delta$ is satisfied, (ii) no further beneficial removal

Algorithm 1 Greedy Margin Mask

Inputs:
Prompt $\boldsymbol{q}$ with rollouts $\{\boldsymbol{\tau}_i\}_{i=1}^N$, outcome rewards $\{R_i\}_{i=1}^N$ with $R_i \in \{0,1\}$,
min retention ratio $\rho \in (0,1)$, trajectory-level returns $\{G_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau}_i)\}_{i=1}^N$,
target margin δ, mode $\in \{\mathrm{MinMax}, \mathrm{Mean}\}$.
Output: Keep-mask $\{k_i\}_{i=1}^N \in \{0,1\}^N$ ▷ $k_i = 1$ *means "keep trajectory* $\boldsymbol{\tau}_i$*".*
Notation:
For any two non-empty subsets $A \subseteq S_+(\boldsymbol{q})$ and $B \subseteq S_-(\boldsymbol{q})$, we define prompt-level margin
$\textsc{Margin}(A, B; \mathrm{MinMax}) = \min_{\boldsymbol{\tau}\in A} G_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau}) - \max_{\boldsymbol{\tau}\in B} G_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau})$,
$\textsc{Margin}(A, B; \mathrm{Mean}) = \mathrm{mean}_{\boldsymbol{\tau}\in A} G_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau}) - \mathrm{mean}_{\boldsymbol{\tau}\in B} G_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau})$,

function GREEDYMARGINMASK($\boldsymbol{q}, \{\boldsymbol{\tau}_i, R_i, G_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau}_i)\}_{i=1}^N, \delta, \rho, \mathrm{mode}$)
  ▷ *Step 1: split the group by outcome correctness.*
  $S_+(\boldsymbol{q}) \leftarrow \{\boldsymbol{\tau}_i \mid R_i = 1\}, \quad S_-(\boldsymbol{q}) \leftarrow \{\boldsymbol{\tau}_i \mid R_i = 0\}$
  $N_+ \leftarrow |S_+(\boldsymbol{q})|, \quad N_- \leftarrow |S_-(\boldsymbol{q})|$ ▷ *number of correct/incorrect trajectories*
  $k_i \leftarrow 1, \quad \forall i = 1, \ldots, N$ ▷ *initialize: keep all trajectories*
  if $N_+ = 0$ **or** $N_- = 0$ **then**
    return $\{k_i\}_{i=1}^N$ ▷ *ordering is not defined; no masking*
  end if

  ▷ *Step 2: sort each side so that the most ordering-violating trajectory is at the front.*
  $L_+ \leftarrow$ sort $S_+(\boldsymbol{q})$ by $G_{\mathrm{OPD}}(\boldsymbol{q},\cdot)$ ascending ▷ $L_+[1]$ = *correct with lowest return*
  $L_- \leftarrow$ sort $S_-(\boldsymbol{q})$ by $G_{\mathrm{OPD}}(\boldsymbol{q},\cdot)$ descending ▷ $L_-[1]$ = *incorrect with highest return*

  ▷ *Step 3: iteratively drop the trajectory whose removal increases the margin the most.*
  while $\textsc{Margin}(L_+, L_-; \mathrm{mode}) < \delta$ **do**
    ▷ *Admissibility: a side may be shortened only while it stays above its retention floor.*
    $c_+ \leftarrow \big(|L_+| > \lceil \rho N_+ \rceil\big), \quad c_- \leftarrow \big(|L_-| > \lceil \rho N_- \rceil\big)$
    if not c_+ **and not** c_- **then**
      break ▷ *minimum retention ratio reached on both sides*
    end if
    $m \leftarrow \textsc{Margin}(L_+, L_-; \mathrm{mode}), \quad \Delta_+ \leftarrow -\infty, \quad \Delta_- \leftarrow -\infty$ ▷ $-\infty$*: inadmissible*
    if c_+ **then**
      $\Delta_+ \leftarrow \textsc{Margin}(L_+ \setminus \{L_+[1]\}, L_-; \mathrm{mode}) - m$ ▷ *drop the worst correct*
    end if
    if c_- **then**
      $\Delta_- \leftarrow \textsc{Margin}(L_+, L_- \setminus \{L_-[1]\}; \mathrm{mode}) - m$ ▷ *drop the best incorrect*
    end if
    if $\max(\Delta_+, \Delta_-) \le 0$ **then**
      break ▷ *no admissible removal can further improve the margin*
    end if
    if $\Delta_+ > \Delta_-$ **then**
      $\boldsymbol{\tau}_{\mathrm{drop}} \leftarrow \textsc{PopFront}(L_+)$ ▷ *greedy drop on the positive side*
    else
      $\boldsymbol{\tau}_{\mathrm{drop}} \leftarrow \textsc{PopFront}(L_-)$ ▷ *greedy drop on the negative side*
    end if
    $k_{\mathrm{idx}(\boldsymbol{\tau}_{\mathrm{drop}})} \leftarrow 0$ ▷ *exclude this trajectory from the subsequent gradient update*
  end while
  return $\{k_i\}_{i=1}^N$
end function

exists, or (iii) a minimum retention ratio $\rho \in (0,1)$ is reached to prevent excessive data loss. The masked trajectories are excluded from the subsequent gradient update by zeroing their token-level advantages, *i.e.,* $\widetilde{A}_t(\boldsymbol{q},\boldsymbol{\tau}_i) = k_i \cdot r_t^{\mathrm{OPD}}(\boldsymbol{\tau}_i)$, where $k_i \in \{0,1\}$ is the keep mask, while every retained trajectory keeps its original token-level rewards. In distributed training, the trajectory-level statistics are aggregated across all ranks via AllReduce so that the masking is deterministic and consistent across devices. The procedure is in algorithm 1.

Algorithm 2 Margin Shift

Inputs:
Prompt $\boldsymbol{q}$ with rollout group $\{\boldsymbol{\tau}_i\}_{i=1}^N$, outcome rewards $\{R_i\}_{i=1}^N$ with $R_i \in \{0,1\}$,
trajectory-level distillation returns $\{G_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau}_i)\}_{i=1}^N$,
target margin δ, mode $\in \{\mathrm{MinMax}, \mathrm{Mean}\}$, direction $\in \{\mathrm{Lift}, \mathrm{Suppress}, \mathrm{Spread}\}$.
Output: Calibrated trajectory-level returns $\{\widetilde{G}_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau}_i)\}_{i=1}^N$

function MARGINSHIFT($\boldsymbol{q}, \{\boldsymbol{\tau}_i, R_i, G_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau}_i)\}_{i=1}^N, \delta$, mode, direction)
  ▷ *Step 1: split the group by outcome correctness.*
  $S_+(\boldsymbol{q}) \leftarrow \{\boldsymbol{\tau}_i \mid R_i = 1\}, \quad S_-(\boldsymbol{q}) \leftarrow \{\boldsymbol{\tau}_i \mid R_i = 0\}$
  if $S_+(\boldsymbol{q}) = \emptyset$ **or** $S_-(\boldsymbol{q}) = \emptyset$ **then**
    return $\{\widetilde{G}_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau}_i) \leftarrow G_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau}_i)\}_{i=1}^N$ ▷ *ordering is not defined*
  end if

  ▷ *Step 2: summarize each side and compute the prompt-level margin $m(\boldsymbol{q})$.*
  if mode $=$ MinMax **then**
    $G_+(\boldsymbol{q}) \leftarrow \min_{\boldsymbol{\tau} \in S_+(\boldsymbol{q})} G_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau})$ ▷ *worst-scoring correct trajectory*
    $G_-(\boldsymbol{q}) \leftarrow \max_{\boldsymbol{\tau} \in S_-(\boldsymbol{q})} G_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau})$ ▷ *best-scoring incorrect trajectory*
  else
    $G_+(\boldsymbol{q}) \leftarrow \mathrm{mean}_{\boldsymbol{\tau} \in S_+(\boldsymbol{q})} G_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau})$ ▷ *average correct score*
    $G_-(\boldsymbol{q}) \leftarrow \mathrm{mean}_{\boldsymbol{\tau} \in S_-(\boldsymbol{q})} G_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau})$ ▷ *average incorrect score*
  end if
  $m(\boldsymbol{q}) \leftarrow G_+(\boldsymbol{q}) - G_-(\boldsymbol{q})$

  ▷ *Step 3: additive correction when the margin is below the target.*
  $\widetilde{G}_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau}_i) \leftarrow G_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau}_i), \quad \forall i = 1,\ldots,N$ ▷ *start from the uncalibrated returns*
  if $m(\boldsymbol{q}) < \delta$ **then**
    $\lambda(\boldsymbol{q}) \leftarrow \delta - m(\boldsymbol{q})$ ▷ *amount by which the margin falls short of δ*
    if direction $=$ Lift **then**
      $\widetilde{G}_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau}) \mathrel{+}= \lambda(\boldsymbol{q}), \quad \forall \boldsymbol{\tau} \in S_+(\boldsymbol{q})$ ▷ *pull all correct trajectories up*
    else if direction $=$ Suppress **then**
      $\widetilde{G}_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau}) \mathrel{-}= \lambda(\boldsymbol{q}), \quad \forall \boldsymbol{\tau} \in S_-(\boldsymbol{q})$ ▷ *push all incorrect trajectories down*
    else
      $\widetilde{G}_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau}) \mathrel{+}= \lambda(\boldsymbol{q})/2, \quad \forall \boldsymbol{\tau} \in S_+(\boldsymbol{q})$ ▷ *split: half up on the positive side, ...*
      $\widetilde{G}_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau}) \mathrel{-}= \lambda(\boldsymbol{q})/2, \quad \forall \boldsymbol{\tau} \in S_-(\boldsymbol{q})$ ▷ *... and half down on the negative side*
    end if
  end if
  return $\{\widetilde{G}_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau}_i)\}_{i=1}^N$
end function

Detailed implementation of margin shift. The margin shift strategy applies a minimal additive correction to the trajectory-level returns so that the margin exactly meets the target δ, rather than discarding any sample. Given the rollout group $\{\boldsymbol{\tau}_i\}_{i=1}^N$ of prompt $\boldsymbol{q}$, we first compute the current margin $m(\boldsymbol{q})$ with the chosen mode (Mean by default). If $m(\boldsymbol{q}) < \delta$, we define the required shift as $\lambda(\boldsymbol{q}) = \delta - m(\boldsymbol{q}) > 0$ and distribute it across trajectories in one of three directions: (i) Lift: add $\lambda(\boldsymbol{q})$ to every positive trajectory, *i.e.,* $\widetilde{G}_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau}) = G_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau}) + \lambda(\boldsymbol{q})\mathbf{1}\{R(\boldsymbol{q},\boldsymbol{\tau}) = 1\}$, which matches Eq. (11) in the main text; (ii) Suppress: subtract $\lambda(\boldsymbol{q})$ from every negative trajectory, *i.e.,* $\widetilde{G}_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau}) = G_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau}) - \lambda(\boldsymbol{q})\mathbf{1}\{R(\boldsymbol{q},\boldsymbol{\tau}) = 0\}$; and (iii) Spread: split the correction symmetrically, adding $\lambda(\boldsymbol{q})/2$ to positives and subtracting $\lambda(\boldsymbol{q})/2$ from negatives. All three variants share the same three properties. First, they preserve the relative ordering within $S_+(\boldsymbol{q})$ and within $S_-(\boldsymbol{q})$, since each side is displaced by a single constant. Second, they bring the calibrated margin to exactly δ under the mode that measures it: with MinMax, the gap between the worst positive and the best negative becomes δ; with Mean, the default in our experiments, it is the gap between the two group means that becomes δ. Third, since the two sides are pulled apart by the same total amount $\lambda(\boldsymbol{q})$ in

all three directions, the score difference of every positive–negative pair grows by exactly $\lambda(\boldsymbol{q})$, so no pair becomes more inverted and the number of ordering violations inside the group cannot increase. The Mean mode therefore reduces the pairwise violations certified by Eq. (9) without necessarily removing all of them, which is the price of letting the two group means, rather than one extreme trajectory, decide the size of the shift. We adopt this trade-off because it is the more stable of the two in practice, as the Mean and MinMax variants in Table E.5 show. In implementation, the correction is added to the token-level rewards of each affected trajectory instead of replacing them, *i.e.*, $\widetilde{A}_t(\boldsymbol{q},\boldsymbol{\tau}) = r_t^{\mathrm{OPD}}(\boldsymbol{\tau}) + \big(\widetilde{G}_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau}) - G_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau})\big)$, so that the fine-grained token-level structure of the supervision is preserved while its trajectory-level aggregation becomes $\widetilde{G}_{\mathrm{OPD}}(\boldsymbol{q},\boldsymbol{\tau})$. In distributed training, the aggregation of trajectory-level statistics and the computation of $\lambda(\boldsymbol{q})$ are done via AllReduce to ensure consistency across devices. The procedure is in algorithm 2.

Handling degenerate groups. A group can still be all-correct or all-wrong, in which case no target ratio is attainable and the prompt-level margin $m(\boldsymbol{q})$ is undefined. Such degenerate groups simply skip the outcome-guided margin calibration, *i.e.*, $\widetilde{A}_t(\boldsymbol{q},\boldsymbol{\tau}_i) = r_t^{\mathrm{OPD}}(\boldsymbol{\tau}_i)$. This keeps every rollout useful for distillation while restricting the outcome-guided correction to the groups where the positive–negative ordering is actually defined.

B.6 Qualitative Visualization of Margin Calibration

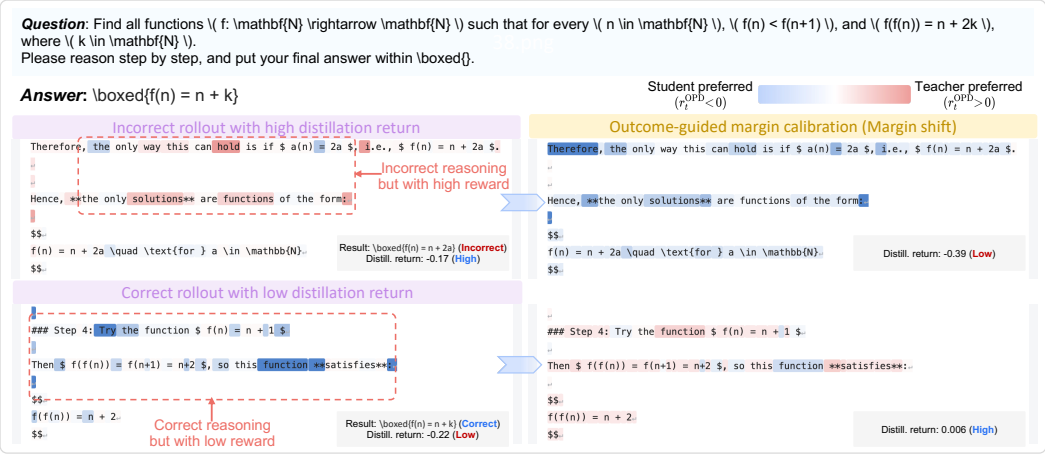


Figure B.3: **Heatmap visualization of failure modes in OPD and the effect of margin shift.** *Left*: an incorrect rollout with a high distillation return (top) and a correct rollout with a low one (bottom). *Right*: the same two rollouts after our margin shift, with the outcome ordering restored.

To complement the algorithmic description above, we provide a token-level visualization of the two failure modes and the effect of margin calibration. As shown in Fig. B.3, we display the two failure modes under the same question: the overestimation of incorrect trajectories (*top-left*) and the underestimation of correct trajectories (*bottom-left*). Each token is colored by its reward value: blue tokens indicate student-preferred ($r_t^{\mathrm{OPD}} < 0$), and red tokens indicate teacher-preferred ($r_t^{\mathrm{OPD}} > 0$), with saturation proportional to magnitude. On the *top-left*, an *incorrect* rollout still accumulates a high distillation return: most of its tokens are saturated red, since they fall on regions where the teacher dominates the student. On the *bottom-left*, a *correct* rollout receives a low distillation return: its tokens are already well-covered by the student, so the teacher provides little additional return (predominantly faint colors with some blue). The *right* column shows the same two rollouts after our outcome-guided margin calibration. Concretely, the per-token rewards are uniformly shifted so that the trajectory-level aggregation aligns with the outcome reward.

C Training Details

In this section, we present details related to training, including the training setup (§C.1), the training pseudocode (§C.2), the training datasets (§C.3), the training reward acquisition (§C.4), the training dynamics (§C.5), and the training complexity analysis (§C.6).

C.1 Training Setup

Training Prompt Template

Math Reasoning

```
<|im_start|>user
{question}
Please reason step by step, and put your final answer within \boxed{}.<|im_end|>
<|im_start|>assistant
```

Code Reasoning

```
<|im_start|>user
{question}
Write Python code to solve the problem. Present the code in
```python
Your code
```
at the end.
You need to think first then write the Python code.<|im_end|>
<|im_start|>assistant
```

Multimodal Math Reasoning

```
<|im_start|>user
<image>
{question}
Please solve the problem step by step and put your answer in one \boxed{}.<|im_end|>
<|im_start|>assistant
```

Models. We conduct experiments on the Qwen3 family (Yang et al., 2025a; Bai et al., 2025a). For textual experiments, we use Qwen3-4B/1.7B student models. In the same-sized setting, we apply domain-specific RL to Qwen3-4B to obtain specialized teachers. In the strong-to-weak setting, we use Qwen3-30B-A3B-Instruct-2507 as the strong teacher. For multimodal experiments, we use Qwen3-VL-2B-Instruct and Qwen3-VL-4B-Instruct as student models, and obtain multimodal teachers through domain-specific RL.

The `Uni-OPD` objective. We instantiate the general multi-teacher objective in Eq. (4) with two choices in our implementation. For the teacher weights $w_i(\boldsymbol{q})$, we use a one-hot teacher assignment: each prompt $\boldsymbol{q}$ belongs to a single domain, and all its sampled trajectories are scored only by the teacher π_{T_i} of that domain, so $w_i(\boldsymbol{q})$ becomes an indicator with $\sum_{i=1}^{M} w_i(\boldsymbol{q}) = 1$. This keeps each trajectory to a single teacher forward pass and lets every expert teacher contribute the strongest capability on its own domain. For the divergence $\mathcal{D}_{\mathrm{KL}}$, we adopt the top-1 (sampled-token) KL (Gu et al., 2023; Agarwal et al., 2024), *i.e.,* the token-level reward uses the teacher–student log-probability difference on the sampled token o_t as in Eq. (3), which avoids materializing the full teacher vocabulary distribution and is widely used in practice to reduce memory overhead (Yang et al., 2026b;c).

To support multi-teacher OPD for both LLMs and MLLMs, we build `Uni-OPD` upon a widely used training framework, Miles[1]. Specifically, we use Megatron-LM[2] (Shoeybi et al., 2019) as the training backend and SGLang[3] as the rollout inference engine. For teacher models, we deploy them as independent SGLang services that can be accessed via HTTP from arbitrary locations to obtain token-level rewards, enabling flexible teacher extensions and scalable multi-teacher integration.

Each teacher is served behind a pool of SGLang endpoints with client-side shuffled round-robin load balancing, and a lightweight task-to-teacher routing table dispatches every prompt to the teacher best matched to its domain (*e.g.,* math reasoning or code generation), so that new teachers or new tasks can be plugged in by simply extending the registry without touching the training loop. Because each teacher only needs to expose its prefill-time `input_token_logprobs`, no gradient, KV

[1] https://github.com/radixark/miles
[2] https://github.com/nvidia/megatron-lm
[3] https://github.com/sgl-project/sglang

cache, or parameter is shared with the student, which keeps teachers fully stateless and decouples their deployment from the trainer. As a result, teacher scoring overlaps with student generation and contributes negligible overhead to the throughput. Training prompts are provided in prompt 1.

General training hyperparameters. All general training settings, including the batch size, rollout numbers, learning rate schedule, optimizer choice, and so on, are identical to those used in ExOPD[4] (Yang et al., 2026b), ensuring a fair and controlled comparison.

RL training setup. Teacher models are trained using reinforcement learning (RL). Detailed training settings of the teacher models are provided in Table C.1.

Table C.1: **Teacher model training configuration with GRPO.**

Group	Setting	Value
Model	Base model	LLM: Math, Code: Qwen3-4B MLLM: Math, Logic, Document: Qwen3-VL-4B-Inst.
	Training steps	LLM: Math, Code: 500, 300 MLLM: Math, Logic, Document: 300, 300, 160
Optimization	Tensor Parallelism (TP)	2
	Micro batch size / GPU	1
	Training batch size	128
	Learning rate	1×10^{-6}
	Warm-up steps	0
	LR schedule	Constant
	ZeRO stage	3
	Optimizer	Adam
Sequence	Max prompt length	2048
	Max response length	16384
RL Algorithm	Advantage estimator	GRPO
	GRPO clip ratio	0.2
	Use KL in reward	False
	KL loss coefficient	0.0
	Entropy coefficient	0.0
Rollout	Samples per prompt (N)	8
	Temperature	1.0
	Top-p	0.95
	Top-k	50
Hardware	GPUs	16× NVIDIA H20

Table C.2: **OPD training configuration.** Most hyperparameters inherit from the teacher RL setup in Table C.1; only the entries that differ between OPD and RL are listed here.

Group	Setting	Textual	Multimodal
Optimization	Training batch size	64	64
	Samples per prompt (N)	16	16
Online filter	Filter mode	Sample filter	Sample filter
	Correct/Incorrect ratio	1:1	1:1
Margin calibration	Scope	Group	Group
	Mode	Mean	Mean
	Direction	SPREAD	LIFT
	Target margin δ	0.4	0

OPD training setup. For OPD, we inherit most hyperparameters (*e.g.,* learning rate, optimizer, and sequence lengths) from the teacher RL setup in Table C.1, so that the student is trained under the same optimization regime as its teachers. The OPD-specific entries, including the training batch size, the number of on-policy samples per prompt, the online correctness-aware filter, and the margin calibration configuration, are summarized in Table C.2.

Concretely, we use a training batch size of 64 and sample $N = 16$ on-policy rollouts per prompt, which we find provides a good trade-off between return estimation quality and computational efficiency (see the ablation in Table E.6). The online correctness-aware filter is applied in sample filter mode with a target correct-to-incorrect ratio of 1:1 within each rollout group, following §A.2.

[4] https://github.com/RUCBM/G-OPD

For margin calibration (§B.5), we adopt *group-level mean* normalization in both domains, while the shift direction and target margin are tuned per domain: for the textual domain, we use Spread with $\delta = 0.4$, and for the multimodal domain, we use Lift with $\delta = 0$.

Algorithm 3 `Uni-OPD`: Outcome-guided Policy Distillation with Margin Calibration

Input: Teacher π_{T}, student $\pi_{\boldsymbol{\theta}}$, dataset $\mathcal{D}$, group size N, batch size B, target margin δ, learning rate η, calibration mode $\in \{\textsc{Mask}, \textsc{Shift}\}$, margin mode $\in \{\mathrm{MinMax}, \mathrm{Mean}\}$, shift direction $\in \{\mathrm{Lift}, \mathrm{Suppress}, \mathrm{Spread}\}$, min retention ratio ρ.
Output: Updated student parameters $\boldsymbol{\theta}$.

function UniOPD($\pi_{\mathrm{T}}, \pi_{\boldsymbol{\theta}}, \mathcal{D}, N, B, \delta, \eta$, calibration mode, margin mode, direction, ρ)
  ▷ ***Offline difficulty-aware data balancing*** *(once before training; see §A.1).*
  $\mathcal{D}_{\mathrm{bal}} \leftarrow$ reweight $\mathcal{D}$ towards a flatter difficulty distribution

  while not converged **do**
    Sample a prompt batch $\mathcal{B} \subset \mathcal{D}_{\mathrm{bal}}$ with $|\mathcal{B}| = B$ ▷ *a fresh batch at every training step*

    ▷ ***Rollout and token-level scoring*** *(per prompt).*
    for all prompt $\boldsymbol{q} \in \mathcal{B}$ **do**
      Rollout N trajectories $\{\boldsymbol{\tau}_i\}_{i=1}^{N} \sim \pi_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{q})$
      for $i = 1, \ldots, N$ **do**
        Obtain outcome reward $R_i = R(\boldsymbol{q}, \boldsymbol{\tau}_i) \in \{0, 1\}$
        for all token $o_t \in \boldsymbol{\tau}_i$ **do**
          $r_t^{\mathrm{OPD}}(\boldsymbol{\tau}_i) \leftarrow \log \pi_{\mathrm{T}}(o_t \mid \boldsymbol{q}, \boldsymbol{o}_{<t}) - \log \pi_{\boldsymbol{\theta}}(o_t \mid \boldsymbol{q}, \boldsymbol{o}_{<t})$ ▷ *token-level OPD reward*
        end for
        ▷ *Trajectory-level distillation return (Eq. (5)).*
        $G_{\mathrm{OPD}}(\boldsymbol{q}, \boldsymbol{\tau}_i) \leftarrow \dfrac{1}{|\boldsymbol{\tau}_i|} \sum_{t=1}^{|\boldsymbol{\tau}_i|} r_t^{\mathrm{OPD}}(\boldsymbol{\tau}_i)$
      end for
      Partition: $S_+(\boldsymbol{q}) \leftarrow \{\boldsymbol{\tau}_i \mid R_i = 1\}, \quad S_-(\boldsymbol{q}) \leftarrow \{\boldsymbol{\tau}_i \mid R_i = 0\}$ ▷ *outcome partition*
    end for

    ▷ ***Online correctness-aware data balancing*** *(within each rollout group; see §A.2).*
    $\{\mathcal{B}(\boldsymbol{q})\}_{\boldsymbol{q} \in \mathcal{B}} \leftarrow \textsc{OnlineBalancing}\big(\mathcal{B}, \{R_i\}_{\boldsymbol{q},i}\big)$ ▷ *retained rollouts of* $\boldsymbol{q}$

    ▷ ***Outcome-guided margin calibration*** *(per prompt; Eqs. (9) and (10)).*
    for all prompt $\boldsymbol{q} \in \mathcal{B}$ **do**
      $\widetilde{G}_{\mathrm{OPD}}(\boldsymbol{q}, \boldsymbol{\tau}_i) \leftarrow G_{\mathrm{OPD}}(\boldsymbol{q}, \boldsymbol{\tau}_i), \quad k_{\boldsymbol{q},i} \leftarrow 1, \quad \forall \boldsymbol{\tau}_i \in \mathcal{B}(\boldsymbol{q})$ ▷ *default: no calibration*
      if $S_+(\boldsymbol{q}) \neq \emptyset$ **and** $S_-(\boldsymbol{q}) \neq \emptyset$ **then**
        $m(\boldsymbol{q}) \leftarrow \textsc{Margin}\big(S_+(\boldsymbol{q}), S_-(\boldsymbol{q})$; margin mode$\big)$ ▷ *Eq. (9);* Margin *as in algorithm 1*
        if calibration mode $=$ Mask **then**
          $\{k_{\boldsymbol{q},i}\} \leftarrow \textsc{GreedyMarginMask}\big(\boldsymbol{q}, \mathcal{B}(\boldsymbol{q}), \delta, \rho$, margin mode$\big)$ ▷ *algorithm 1*
        else
          $\{\widetilde{G}_{\mathrm{OPD}}(\boldsymbol{q}, \boldsymbol{\tau}_i)\} \leftarrow \textsc{MarginShift}\big(\boldsymbol{q}, \mathcal{B}(\boldsymbol{q}), \delta$, margin mode, direction$\big)$ ▷ *algorithm 2*
        end if
      else
        ▷ *Degenerate group: $m(\boldsymbol{q})$ is undefined, so calibration is skipped (see §A.2).*
      end if
    end for

    ▷ ***Token-level advantages and policy update.***
    for all prompt $\boldsymbol{q} \in \mathcal{B}$, rollout $\boldsymbol{\tau}_i \in \mathcal{B}(\boldsymbol{q})$ **do**
      $\Delta_{\boldsymbol{q},i} \leftarrow \widetilde{G}_{\mathrm{OPD}}(\boldsymbol{q}, \boldsymbol{\tau}_i) - G_{\mathrm{OPD}}(\boldsymbol{q}, \boldsymbol{\tau}_i)$ ▷ *zero if $\boldsymbol{\tau}_i$ was left uncalibrated*
      for all token $o_t \in \boldsymbol{\tau}_i$ **do**
        $\widetilde{A}_t(\boldsymbol{q}, \boldsymbol{\tau}_i) \leftarrow k_{\boldsymbol{q},i} \cdot \big(r_t^{\mathrm{OPD}}(\boldsymbol{\tau}_i) + \Delta_{\boldsymbol{q},i}\big)$ ▷ *uniform shift on every token reward*
      end for
    end for
    $\mathcal{L}(\boldsymbol{\theta}) = -\,\mathbb{E}_{\boldsymbol{q},\boldsymbol{\tau}_i,t}\Big[\widetilde{A}_t(\boldsymbol{q}, \boldsymbol{\tau}_i)\, \log \pi_{\boldsymbol{\theta}}(o_t \mid \boldsymbol{q}, \boldsymbol{o}_{<t})\Big]$
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta})$ ▷ *one gradient step on the student*
  end while
  return $\boldsymbol{\theta}$
end function

C.2 Training Pseudocode

The full training procedure of `Uni-OPD` is summarized in algorithm 3. In brief, the procedure (1) samples a prompt batch with offline difficulty-aware balancing (§A.1); (2) rolls out N trajectories per prompt and computes the trajectory-level distillation return G_{OPD} from teacher–student log-probability differences (Eq. (5)); (3) applies online correctness-aware balancing within each rollout group (§A.2); (4) calibrates G_{OPD} via the prompt-level margin $m(\boldsymbol{q})$ (Eq. (9)) using either Greedy Margin Mask (algorithm 1) or Margin Shift (algorithm 2); and (5) turns the calibration into token-level advantages, either by uniformly shifting the token-level rewards of a trajectory or by dropping the masked rollouts, and updates the student $\pi_{\boldsymbol{\theta}}$.

C.3 Training Data

Textual math reasoning data. We use a subset of the DeepMath dataset (He et al., 2025b) with difficulty level ≥ 6 to train mathematical reasoning ability, comprising 57.0K samples.

Textual code generation data. We use the Code subset of the Eurus-2-RL-Data dataset (Cui et al., 2025) with 25.3K samples to train code generation ability.

Multimodal math reasoning data. For multimodal math reasoning, we draw 14.8K samples from the OpenMMReasoner dataset[5], covering MMK12, WeMath-Standard, and WeMath-Pro subsets.

Multimodal logic reasoning data. We collect 14.8K samples spanning AlgoPuzzle, PuzzleVQA, and ThinkLite-VL-Hard subsets from the OpenMMReasoner-RL-74K dataset.

Multimodal document understanding data. We include 14.6K document understanding samples, obtained by 15% sampling from the TQA subset of OpenMMReasoner with ChartQA (Masry et al., 2022) and InfographicsVQA (Mathew et al., 2022) training sets.

C.4 Training Reward Acquisition

In this section, we describe how training rewards are obtained for different data sources. For textual math reasoning tasks, we use the rule-based verifier provided by DeepMath[6] to determine whether generated answers are correct. For textual code generation tasks, we use the rule-based verifier provided by PRIME[7] to evaluate the correctness of generated code. For multimodal tasks, we use the verifier released by OpenMMReasoner[8] to assess whether generated answers are correct.

C.5 Training Dynamics

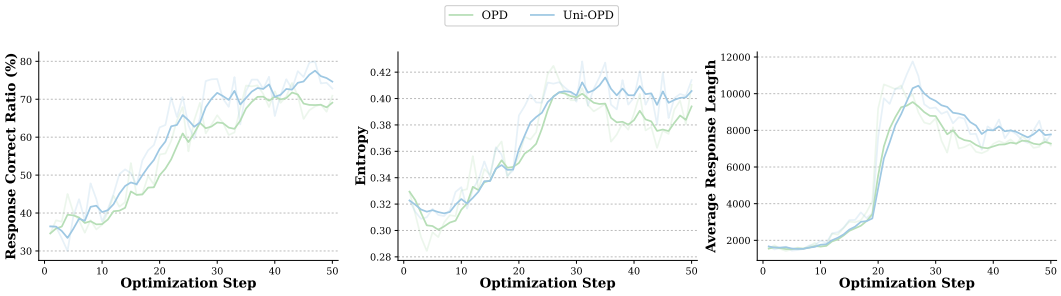


Figure C.1: **Training dynamics of OPD and `Uni-OPD` for multi-teacher distillation.** We track three indicators along the optimization trajectory: response correctness (%), policy entropy, and average response length.

[5] https://huggingface.co/datasets/OpenMMReasoner/OpenMMReasoner-RL-74K
[6] https://github.com/zwhe99/DeepMath
[7] https://github.com/PRIME-RL/PRIME
[8] https://github.com/EvolvingLMMs-Lab/OpenMMReasoner

Fig. C.1 demonstrates the effectiveness of Uni-OPD along three complementary axes. From a comparable starting point ($\sim$35% correct, entropy $\sim$0.33, length $\sim$1.6k), Uni-OPD converges to a substantially higher response-correct ratio than OPD, peaking at 80.6% versus 75.2% and averaging 75.5% over the final 10 steps versus OPD's 69.1% (+6.4 absolute points). Crucially, this accuracy gain is not obtained by sacrificing exploration: policy entropy rises mildly under both methods, with Uni-OPD maintaining a marginally higher steady-state value, ruling out the entropy-collapse failure mode that typically plagues teacher-driven training. Meanwhile, the average response length grows from $\sim$1.6k to $\sim$8k tokens, with Uni-OPD producing slightly longer outputs than OPD (7.8k vs. 7.1k), indicating that the model learns to perform more elaborate reasoning rather than collapsing to short, high-confidence shortcuts. Together, these trends suggest that Uni-OPD provides a consistent improvement over OPD without adverse effects on exploration or response length.

C.6 Training Complexity

Beyond vanilla OPD, Uni-OPD introduces lightweight components on top of the standard per-iteration cost during training: *online correctness-aware data balancing* (per batch; §A.2), and *outcome-guided margin calibration* via Margin Mask / Shift (per prompt; §B.5). Let B be the training batch size (number of prompts) and N be the rollout group size. The online balancing only subsamples trajectories inside each rollout group based on their precomputed $\{R_i\}$, costing $O(BN)$ per iteration. Margin Mask and Margin Shift both operate on the N trajectory-level returns within each prompt group: Margin Shift is $O(N)$ per prompt, while the greedy variant of Margin Mask is at most $O(N^2)$ per prompt in the worst case (typically $N \leq 16$ in our setup).

In contrast, the dominant per-iteration cost of OPD comes from two stages whose complexity scales linearly with the total number of rollout tokens $T_{\text{tok}} = \sum_{i=1}^{BN} |\boldsymbol{\tau}_i|$ and quadratically with the hidden size d: (i) sampling BN on-policy rollouts from the student, and (ii) running a teacher prefill pass over these rollouts to obtain token-level log-probabilities, each of order $O(T_{\text{tok}}\, d^2)$ for transformer forward passes. Typical numbers in our setup ($B = 64$, $N = 16$, average length $\sim$8k) give T_{tok} on the order of 8×10^6 tokens per iteration. All of Uni-OPD's additional computation scales with the number of trajectories rather than the number of tokens, involves only scalar comparisons and additions, and is therefore several orders of magnitude cheaper than the rollout and teacher-scoring stages that OPD already pays. In practice, we observe that enabling all three components adds less than 1% wall-clock overhead per iteration relative to vanilla OPD, while delivering the accuracy improvements reported in §C.5 and the main experiments. Thus Uni-OPD offers a favorable accuracy–compute trade-off: a negligible compute surcharge in exchange for consistently better final performance.

D Evaluation Details

D.1 Evaluation Benchmarks

We evaluate our Uni-OPD on a comprehensive benchmark suite spanning textual and multimodal capabilities, organized along five capability axes:

- **Textual Math Reasoning:**
  - AIME (2024/2025) (AI-MO, 2024; OpenCompass, 2025): A prestigious high school mathematics competition featuring challenging problems that test deep mathematical reasoning.
  - HMMT25 (Feb & Nov) (Balunović et al., 2025): Contest-level benchmarks designed to evaluate advanced reasoning across algebra, geometry, combinatorics, and other domains.
- **Textual Code Generation:**
  - HumanEval+ (Liu et al., 2023b): A set of 164 programming problems evaluating functional correctness, covering language understanding, reasoning, algorithms, and basic mathematics.
  - MBPP+ (Liu et al., 2023b): A collection of $\sim$1,000 crowd-sourced Python tasks targeting entry-level programming skills, including fundamentals and standard library usage.
  - LiveCodeBench (v6) (Jain et al., 2024): A contamination-free and continuously updated benchmark assessing not only code generation but execution, self-repair, and test prediction.
- **Multimodal Math Reasoning:**

- MathVision (Wang et al., 2024a): A curated dataset of 3,040 visual problems sourced from real competitions, spanning 16 disciplines and multiple difficulty levels for evaluating multimodal mathematical reasoning.
- DynaMath (Zou et al., 2025): A dynamically generated benchmark based on 501 seed question generators, enabling diverse and scalable evaluation through multiple sampled variants.
- WeMath (Qiao et al., 2025): A large-scale benchmark with 6.5K visual math problems organized into 67 hierarchical knowledge concepts, designed to analyze problem-solving processes.

- **Multimodal Logic Reasoning:**
  - LogicVista (Xiao et al., 2024): A benchmark for evaluating logical reasoning across 5 task types and 9 capabilities using annotated multiple-choice questions with human reasoning.
  - VisuLogic (Xu et al., 2025): A visual benchmark focusing on reasoning directly from visual inputs, with tasks that are difficult to express textually and expose gaps in current MLLMs.
- **Document Understanding:**
  - AI2D (Kembhavi et al., 2016): A diagram understanding benchmark focusing on parsing diagram structure and reasoning over relationships between components via question answering.
  - ChartQA (Masry et al., 2022): A benchmark for question answering over charts, requiring complex visual and logical reasoning over both chart structure and underlying data.
  - DocVQA (Mathew et al., 2021): A large-scale document visual question answering dataset over document images, emphasizing structural and textual understanding.
  - InfoVQA (Mathew et al., 2022): A benchmark on infographic understanding that requires joint reasoning over layout, text, and visual elements with an emphasis on multi-step reasoning.

D.2 Baselines

We compare `Uni-OPD` against several representative baselines for LLM distillation: (1) **SFT**, which performs supervised fine-tuning on teacher-generated trajectories via cross-entropy loss; (2) **ExPO** (Zheng et al., 2025b), a weight-space extrapolation method that merges domain-specific teachers and extrapolates their weights relative to the student model; (3) **ExOPD** (Yang et al., 2026b), a reward-level extrapolation approach that scales the reward factor (> 1) to enable the student to surpass the performance boundaries of its teachers. For strong-to-weak distillation, we further include **IW-OPD** (Xie et al., 2026), which reweights token-level OPD supervision according to cumulative teacher–student discrepancies along each rollout, emphasizing teacher-compatible prefixes and downweighting later drifted tokens. For comparison with a Qwen3-1.7B student, we include **Privileged-OPD** (Kaur et al., 2026) (results reported by He et al. (2026)). For MLLM experiments, we use **vanilla OPD** as the primary baseline.

D.3 Evaluation Setup

Textual evaluations. For all textual evaluations, we use a sampling temperature of 1.0, top-p of 1.0, a maximum generation length of 16,384 tokens, and a fixed random seed of 42. We use the vLLM inference engine to perform sampling. For math reasoning benchmarks, we sample $N = 32$ solutions per problem, while for code generation benchmarks, we sample $N = 4$ solutions per problem. For evaluation, we adopt `Math-Verify`[9] as a rule-based verifier for math reasoning tasks. For code generation, we use the `EvalPlus`[10] and `LiveCodeBench`[11] frameworks to assess functional correctness. For all main results, we report the average accuracy across sampled solutions (*i.e.,* pass@1).

Multimodal evaluations. For multimodal evaluations, we adopt the widely used `LMMs-Eval`[12] (Zhang et al., 2025a) framework and strictly follow its official evaluation protocols and configurations. The reported evaluation metrics are summarized in Table D.1.

[9] https://github.com/huggingface/Math-Verify
[10] https://github.com/evalplus/evalplus
[11] https://github.com/livecodebench/livecodebench
[12] https://github.com/evolvinglmms-lab/lmms-eval

Table D.1: **Evaluation metrics for benchmark datasets.** We summarize the primary metrics used for performance reporting across math, logic, and document understanding tasks.

Category	Tasks	Filter	N-Shot	Reported Metric
Multimodal Math Reasoning	MathVision Test	none	0	mathvision_standard_eval
	DynaMath Reasoning	none	0	dynamath_average
	WeMath TestMini Reasoning	none	0	acc_score
Multimodal Logic Reasoning	LogicVista Reasoning	none	0	acc_score
	LogicVista Reasoning	none	0	format_score
	VisuLogic	none	0	visulogic_acc
Document Understanding	AI2D	flexible-extract	0	exact_match
	ChartQA	none	0	relaxed_human_split
	DocVQA Val	none	0	anls
	InfoVQA Val	none	0	anls

E Further Evaluations

E.1 More Evaluation Results

Table E.1: **Performance of *Qwen3-1.7B Student* under math reasoning and code generation benchmarks.** Teacher models (*i.e., Qwen3-4B-Math-RL* and *Qwen3-4B-Code-RL*) are developed through domain-specific RL. For Privileged-OPD, "–" denotes results not reported in the paper.

Method	Math Reasoning					Code Generation			
	AIME 2024	AIME 2025	HMMT 25 Feb.	HMMT 25 Nov.	**Avg.**	Human Eval+	MBPP+	LCB	**Avg.**
Student (1.7B)	13.9	11.1	5.6	4.9	8.9	61.9	53.4	11.9	42.4
Teacher (RL)	60.1	55.1	32.5	38.5	46.6	85.2	69.8	26.6	60.5
Single–Teacher Distillation									
OPD	42.3	**35.4**	18.4	19.1	28.8	71.8	58.2	26.7	52.2
Privileged-OPD	39.8	30.4	–	–	–	–	–	–	–
`Uni-OPD`	**42.6**	35.1	**20.8**	**20.9**	**29.9**	**73.0**	**60.0**	**28.1**	**53.7**
Multi–Teacher Distillation									
OPD	40.3	32.4	**20.0**	**20.3**	28.3	**73.2**	59.1	25.7	52.7
`Uni-OPD`	**44.0**	**35.1**	19.5	19.8	**29.6**	72.9	**60.5**	**28.0**	**53.8**

Table E.2: **Performance of *Qwen3-VL-2B-Instruct Student* under math reasoning, logic reasoning, and document understanding benchmarks.** Teacher models (*i.e., Qwen3-VL-4B-Instruct-Math-RL*, *Qwen3-VL-4B-Instruct-Logic-RL* and *Qwen3-VL-4B-Instruct-Document-RL*) are developed through domain-specific RL.

Method	Math Reasoning				Logic Reasoning				Document Understanding				
	Math Vision	Dyna Math	We Math	**Avg.**	LogicVista Accuracy	LogicVista Format	Visu Logic	**Avg.**	AI2D	Chart QA	Doc VQA	Info VQA	**Avg.**
Student (2B)	11.1	49.1	48.6	36.3	32.4	59.1	6.4	32.6	73.4	66.1	92.8	72.4	76.2
Teacher (RL)	47.2	65.3	79.5	64.0	52.5	73.8	27.4	51.2	82.5	76.4	95.1	81.6	83.9
Single–Teacher Distillation													
OPD	24.4	54.5	64.8	47.9	35.3	61.6	26.8	41.2	76.1	66.0	**93.0**	72.2	76.8
`Uni-OPD`	**25.5**	**55.2**	**65.0**	**48.6**	**36.8**	**65.2**	**27.6**	**43.2**	**76.7**	**66.6**	92.9	**72.6**	**77.2**
Multi–Teacher Distillation													
OPD	15.2	50.2	57.6	41.0	38.0	65.2	**27.2**	43.4	**76.2**	66.1	92.9	72.5	76.9
`Uni-OPD`	**18.7**	**51.2**	**58.7**	**42.9**	**42.0**	**69.8**	27.0	**46.3**	76.0	**66.5**	**93.0**	**72.6**	**77.0**

Single-teacher and multi-teacher distillation on LLMs and MLLMs. We further evaluate `Uni-OPD` under both single-teacher and multi-teacher distillation settings on LLMs and MLLMs. As shown in Tables E.1 and E.2, our method consistently outperforms the standard OPD baseline across all domains and settings. On the LLM student (*i.e., Qwen3-1.7B*), `Uni-OPD` improves the average scores on both math reasoning and code generation under single-teacher and multi-teacher

distillation. On the MLLM student (*i.e., Qwen3-VL-2B-Instruct*), it delivers consistent gains across math reasoning, logic reasoning, and document understanding. Further, it narrows the gap to the teacher ensemble under multi-teacher distillation. Consistent improvements in smaller students provide strong empirical evidence for our dual-perspective approach, confirming that student exploration and teacher reliability are indeed the fundamental drivers of reliable distillation.

Cross-modal distillation on code generation and logic reasoning. Beyond the cross-modal distillation on math reasoning and code generation, we further conduct cross-modal distillation on code generation and logic reasoning. Specifically, we combine text-only code data with multimodal logic reasoning data, and jointly distill from two domain-specific teachers (*Qwen3-VL-4B-Instruct-Code-RL* and *Qwen3-VL-4B-Instruct-Logic-RL*) into a single *Qwen3-VL-4B-Instruct* student. As shown in Table E.3, `Uni-OPD` outperforms the standard OPD baseline on both the code generation and logic reasoning averages, with the largest gain on LCB (39.6 → 41.4) and LogicVista Accuracy (50.0 → 52.0). These results confirm that `Uni-OPD` integrates heterogeneous text-only and multimodal data under a single training run, further supporting its applicability to cross-modal distillation.

Table E.3: **Performance of *Qwen3-VL-4B-Instruct Student* under code generation and logic reasoning benchmarks.** Teacher models (*i.e., Qwen3-VL-4B-Instruct-Code-RL* and *Qwen3-VL-4B-Instruct-Logic-RL*) are developed through domain-specific RL.

Method	Code Generation				Logic Reasoning			
	Human Eval+	MBPP+	LCB	**Avg.**	LogicVista Accuracy	LogicVista Format	Visu Logic	**Avg.**
Student (4B)	76.8	70.0	37.0	61.3	49.9	66.4	25.1	47.1
Teacher (RL)	82.2	70.5	40.1	64.3	52.5	73.8	27.4	51.2
Multi–Teacher Distillation								
OPD	79.0	68.5	39.6	62.4	50.0	69.3	27.3	48.9
`Uni-OPD`	**79.4**	**69.2**	**41.4**	**63.3**	**52.0**	**73.8**	**28.0**	**51.3**

E.2 Downstream Task Evaluation

Evaluation on general capabilities. To assess the impact of OPD on general downstream performance of the policy model, we evaluate the models on a diverse set of benchmarks from the Hugging Face Open LLM Leaderboard (Beeching et al., 2023) following recent studies (Peng et al., 2025a; Meng et al., 2024). Specifically, we report results on MMLU (Hendrycks et al., 2021), ARC (Clark et al., 2018), HellaSwag (Zellers et al., 2019), TruthfulQA (Lin et al., 2022), WinoGrande (Sakaguchi et al., 2021), GSM8K (Cobbe et al., 2021), CommonsenseQA (Talmor et al., 2019), and IFEval (Zhou et al., 2023b). We strictly follow the standard evaluation protocols provided by the lm-evaluation-harness system[13]. For IFEval, we report the inst_level_loose_acc.

The results are presented in Table E.4. Overall, `Uni-OPD` not only outperforms OPD and domain-specific teachers on math reasoning and code generation benchmarks demonstrated in the main text, but also retains strong performance across a wide range of downstream tasks. These results suggest that OPD serves as a general and effective framework for improving LLM performance beyond task-specific settings.

Table E.4: **General downstream task performance.** Evaluation on 8 general benchmarks to ensure general-purpose capabilities are maintained after OPD.

Model	MMLU	ARC	HellaSwag	TruthfulQA	WinoGrande	GSM8K	CommonsenseQA	IFEval	**Avg.**
Qwen3-4B	68.3	80.7	68.4	54.8	66.6	84.2	75.8	88.9	73.5
Math Teacher (RL)	68.4	80.8	68.5	54.3	66.0	86.7	75.4	89.2	73.7
Code Teacher (RL)	68.3	80.2	68.3	54.8	65.7	85.8	75.7	89.7	73.6
OPD	68.3	80.3	68.4	54.6	66.5	88.6	75.5	89.2	73.9
`Uni-OPD`	68.3	80.3	68.3	54.6	66.0	88.6	75.7	89.2	73.9

[13] https://github.com/EleutherAI/lm-evaluation-harness

E.3 Further Ablation

Hyperparameter analysis for margin shift. As shown in Table E.5, we compare four variants of margin shift against the OPD baseline across math reasoning and code generation benchmarks. The shift scope (Global vs. Group), normalization mode (Mean vs. MinMax), and shift direction (Lift vs. Spread) are ablated systematically. All shift variants consistently outperform the vanilla OPD baseline, demonstrating the general effectiveness of margin shift. Among the variants, *Group + Mean + Spread* achieves the best average performance on both code generation (62.4) and math reasoning (48.2), indicating that group-level mean normalization with bidirectional shifting provides a more calibrated return signal. Since the four variants differ in more than one factor at a time, we read this as evidence for the combination as a whole rather than for the scope, the mode, or the direction in isolation. The observed trend is nonetheless consistent with our expectation that MinMax-based normalization and global-scope statistics are more susceptible to outlier return values, since extreme returns within a batch can distort the shift magnitude and destabilize training, whereas group-level mean normalization produces more robust and consistent return estimates.

Table E.5: **Effectiveness validation of margin shift across different hyperparameters.** We conduct single-teacher distillation experiments with a *Qwen3-4B Student* using individual math and code teachers.

Configuration	Math Reasoning					Code Generation			
	AIME 2024	AIME 2025	HMMT 25 Feb.	HMMT 25 Nov.	**Avg.**	Human Eval+	MBPP+	LCB	**Avg.**
OPD (no shift)	57.9	52.4	30.2	37.8	44.6	82.6	68.8	25.7	59.0
Global + Mean + Lift	61.8	55.2	34.8	39.4	47.8	85.7	71.4	25.7	60.9
Global + MinMax + Lift	62.4	57.3	32.2	38.2	47.5	85.8	71.8	26.7	61.4
Group + MinMax + Spread	63.4	56.7	33.4	39.0	48.1	86.9	70.6	26.7	61.4
Group + Mean + Spread (ours)	62.7	56.3	34.4	39.2	**48.2**	88.3	72.3	26.7	**62.4**

Hyperparameter analysis for rollout number N. As shown in Table E.6, we ablate the rollout number N in OPD while keeping the global batch size fixed at 1024 (*i.e.,* $N \times bs = 1024$), so that increasing N comes at the cost of a smaller per-step batch size bs. For the OPD baseline, performance remains largely stable across all values of N (44.3–44.6 avg.), suggesting that the base method is relatively insensitive to this trade-off. In contrast, OPD with margin shift benefits notably from larger rollout groups: average performance improves from 45.3 at N=4 to 48.2 at N=16, as more responses per prompt yield more reliable relative return estimation for the margin-based calibration. We find that increasing N from 16 to 32 yields comparable performance. Considering return estimation quality, training stability, and computational efficiency, we therefore set N=16 as our default.

Table E.6: **The effects of rollout number.** The global batch size is fixed at $N \times bs = 1024$ throughout.

Method	AIME 2024	AIME 2025	HMMT 25 Feb.	HMMT 25 Nov.	**Avg.**
Student (4B)	23.0	19.3	12.3	9.2	15.9
OPD					
$N = 4, bs = 256$	59.1	54.1	31.5	32.6	44.3
$N = 8, bs = 128$	59.8	52.9	29.6	35.8	44.5
$N = 16, bs = 64$	57.9	52.4	30.2	37.8	44.6
$N = 32, bs = 32$	58.3	51.2	30.6	36.9	44.3
OPD + Margin shift					
$N = 4, bs = 256$	57.9	52.4	33.2	37.8	45.3
$N = 8, bs = 128$	62.5	55.4	31.9	39.2	47.3
$N = 16, bs = 64$	62.7	56.3	34.4	39.2	48.2
$N = 32, bs = 32$	63.1	55.4	34.2	39.6	48.1

E.4 Analysis and Takeaways

Based on our comprehensive and systematic study on both LLMs and MLLMs across single-teacher, multi-teacher, strong-to-weak, and cross-modal distillation settings, we deliver four takeaways to further advance OPD.

- **Balancing reasoning capability and efficiency.** `Uni-OPD` achieves the best performance with substantially fewer optimization steps than RL (Fig. 1), and consistently delivers strong reasoning capability across diverse domains (Tables 1 to 4, and Tables E.1 to E.3).

- **Teacher value comes from the capability gap, not absolute strength alone.** In OPD, even with the same 4B backbone, a domain-specific RL teacher injects new capabilities and knowledge that drive the student to improve and even surpass the teacher (Tables 1 and 2). Moreover, our dual-perspective recipe further translates this gap into student gains, consistently boosting performance across all model sizes.
- **OPD jointly consolidates textual and multimodal capabilities.** Trained jointly on textual and multimodal data, the multimodal student under `Uni-OPD` improves textual code generation and multimodal math/logic reasoning (Tables 4 and E.3). In these settings the per-token signal transfers across both data types, enabling a unified, single-stage path that enhances textual and multimodal reasoning within one multimodal model.
- **OPD cleanly merges specialized capabilities, with related ones reinforcing each other.** Beyond two teachers, `Uni-OPD` extends to three, jointly improving all capabilities (Tables 2 and E.2). OPD thus offers a scalable path for merging many specialists into one reasoner, with related ones synergizing via shared reasoning structure.

F Related Work

F.1 Multimodal Large Language Models

Large Language Models (LLMs) have undergone rapid development in recent years (Touvron et al., 2023; Achiam et al., 2023; AI@Meta, 2024b; Hurst et al., 2024; Yang et al., 2024a; AI@Meta, 2024a; Yang et al., 2025a; Brown et al., 2020; Team et al., 2024; Anthropic, 2023b;a; 2024; Liu et al., 2024a; Guo et al., 2025a; Li et al., 2025), significantly improving reasoning capabilities. Meanwhile, MLLMs have also seen substantial progress (Radford et al., 2021; Shao et al., 2024a; Wang et al., 2025; Tian et al., 2019; Liu et al., 2024e; Yang et al., 2024b; Peng et al., 2025a; Team et al., 2025). Leveraging advances in LLMs, multimodal large language models (MLLMs) further integrate visual and textual representations through cross-modal learning, achieving strong multimodal understanding and generation capabilities. A key driver of this success lies in the combination of large-scale self-supervised pre-training on diverse corpora and subsequent high-quality supervised fine-tuning (SFT), which enables LLMs and MLLMs to exhibit strong generalization and emergent capabilities in real-world tasks (Wang et al., 2024b; Bai et al., 2023; 2025b; Liu et al., 2023a; 2024b;c; Dai et al., 2023; OpenAI, 2023; Zhu et al., 2024; Qu et al., 2025; Yang et al., 2023b; Zhong et al., 2025; Yang et al., 2023a; 2025b; Lai et al., 2024; Peng et al., 2025b; Hou et al., 2026; Peng et al., 2026). Building upon these foundations, KD has emerged as an important paradigm for transferring sophisticated reasoning capabilities from teacher models to more efficient students. Among various distillation strategies, OPD has recently emerged as a mainstream post-training paradigm for both LLMs and MLLMs. In the on-policy setting, however, the effectiveness of distillation is tied to both the quality of student exploration and the reliability of teacher feedback. In this work, we present a dual-perspective optimization strategy from both the student and teacher sides to improve data suitability and training stability in OPD.

F.2 Reinforcement Learning

By optimizing trajectories sampled from the current policy, on-policy RL alleviates distribution mismatch and is often instantiated with verifiable or outcome-based rewards in reasoning tasks. Notable methods include GRPO (Shao et al., 2024b) for critic-free grouped optimization, DAPO (Yu et al., 2026) for decoupled clipping and dynamic sampling, GSPO (Zheng et al., 2025a) for sequence-level stable optimization, and SAPO (Gao et al., 2025) for soft adaptive policy optimization. On-policy RL has achieved broad progress across a wide range of domains (Shao et al., 2024b; Team et al., 2025; Wen et al., 2025; Liu et al., 2025; Zhao et al., 2025; Li et al., 2026a). Recently, some works have also combined RLVR with OPD, such as Self-Distilled RLVR (Yang et al., 2026a) and OpenClaw-RL (Wang et al., 2026). In our work, we use GRPO to obtain stronger domain-specific teachers and use the corresponding reward models as global guidance for return calibration in OPD.

F.3 On-Policy Distillation

Early OPD work, such as MiniLLM (Gu et al., 2023) and GKD (Agarwal et al., 2024), establishes the basic paradigm of using teacher feedback on student-generated trajectories under a reverse KL ob-

jective. Recent studies further broaden this paradigm from multiple perspectives. In self-distillation methods, OPSD (Zhao et al., 2026b) uses privileged information; SDFT (Shenfeld et al., 2026) allows the student to absorb knowledge from retrieved demonstrations while reducing forgetting. SDPO (Hübotter et al., 2026) treats the current model itself as a self-teacher; OPCD (Ye et al., 2026) internalizes context knowledge into model parameters by minimizing reverse KL between the student and a context-conditioned teacher on the student's trajectories. Regarding teacher access, black-box OPD (Ye et al., 2025) introduces a discriminator-guided framework that does not require teacher logits. Several works also focus on improving optimization and efficiency. ExOPD (Yang et al., 2026b) reformulates OPD as weighted dense RL; Fast and Effective OPD (Zhang et al., 2026a) improves efficiency through prefix-only distillation; KDFlow (Zhang et al., 2026b) provides an extensible distillation framework supporting both off-policy and on-policy training; MiMo-V2-Flash (Xiao et al., 2026) introduces multi-teacher OPD, enabling effective capability merging across domains. Revisiting OPD (Fu et al., 2026) identifies three failure modes in sampled-token OPD and proposes teacher top-K local support matching to improve training stability. CoPD (Gu et al., 2026) interleaves branch-specific RLVR with bidirectional mutual OPD, keeping parallel experts behaviorally close enough for effective capability transfer. Li et al. (2026c) rethink OPD in terms of its phenomenology, mechanisms, and training recipes.

Recently, OPD has also begun to extend beyond text-only settings. VOLD (Bousselham et al., 2025) transfers reasoning ability from text teachers to vision-language students through a two-stage pipeline that combines cold-start alignment with GRPO and OPD. Video-OPD (Li et al., 2026b) adapts OPD for long-video grounding and introduces a curriculum to filter unreliable teacher signals. X-OPD (Cao et al., 2026) further extends OPD to speech through cross-modal alignment. In contrast, our work focuses on developing a unified OPD framework with an open recipe for both LLMs and MLLMs.

G Case Studies

We provide qualitative case studies of `Uni-OPD`, standard OPD, and the Student model across both LLM and MLLM benchmarks, covering textual math reasoning, code generation, logical reasoning, multimodal math reasoning, and chart understanding.

We first revisit the math reasoning case in Fig. G.1, and provide a detailed output comparison of standard OPD and our `Uni-OPD`. Standard OPD assigns *high* returns to incorrect trajectories and *low* returns to correct ones. Furthermore, the code generation case in Fig. G.2 highlights `Uni-OPD`'s ability to balance algorithmic efficiency and code readability. These case studies demonstrate how our dual-perspective optimization–specifically by restoring order consistency through margin calibration–leads to more reliable and high-quality model outputs.

Across the multimodal case studies in Figs. G.3 to G.9, our observations reveal three consistent patterns: (a) `Uni-OPD` demonstrates superior efficiency on complex reasoning problems, producing more concise outputs while maintaining correctness, whereas the Student model and standard OPD frequently generate excessively long responses that are truncated before reaching a final answer; (b) `Uni-OPD` achieves higher correctness than the Student model, often succeeding on questions where the Student model fails; and (c) Our data-balancing strategies encourage exploration of informative student-generated states during training, improving `Uni-OPD`'s ability to tackle challenging visual and mathematical reasoning problems that the Student model cannot solve on its own.

Question: Find all functions \(f: \mathbf{N} \rightarrow \mathbf{N} \) such that for every \(n \in \mathbf{N} \), \(f(n) < f(n+1) \), and \(f(f(n)) = n + 2k \), where \(k \in \mathbf{N} \).
Please reason step by step, and put your final answer within \boxed{}.

Answer: \boxed{f(n) = n + k}

OPD solution 1: … (Omitted for brevity.)
Step 5: Are there **other** solutions?
Suppose $ f $ is not of the form $ f(n) = n + 2a $. Let's suppose that it is **strictly increasing**, $ f(n) \in \mathbb{N} $, and such that $ f(f(n)) = n + 2k $, for some $ k \in \mathbb{N} $.
Suppose that $ f(n) = n + a(n) $, where $ a(n) $ is a strictly increasing function (since $ f(n) = n + a(n) $ and $ f $ is strictly increasing, then $ a(n) $ must be strictly increasing).
Then $ f(f(n)) = f(n + a(n)) = n + a(n) + a(n + a(n)) $
So:
$$
f(f(n)) = n + a(n) + a(n + a(n)) = n + 2k \Rightarrow a(n) + a(n + a(n)) = 2k
$$
This is only possible if $ a(n) $ is **constant**. Because otherwise the left-hand side is not consistent across $ n $. So unless $ a(n) = 2a $, a constant, the right-hand side is not of the form $ 2k $, but depends on both $ n $ and $ a(n) $.
Therefore, the only way this can hold is if $ a(n) = 2a $, i.e., $ f(n) = n + 2a $.

Incorrect reasoning but with high return

Hence, **the only solutions** are functions of the form:

$$
f(n) = n + 2a \quad \text{for } a \in \mathbb{N}
$$

… (Omitted for brevity.)

Final Answer:
$$
\boxed{f(n) = n + 2a \text{ for some } a \in \mathbb{N}}
$$
All such functions are of the form $ f(n) = n + 2a $, for some positive integer $ a $.<|im_end|>

OPD solution 2: … (Omitted for brevity.)
Let's try a known function that satisfies this condition.
Step 4: Try the function $ f(n) = n + 1 $

Then $ f(f(n)) = f(n+1) = n+2 $, so this function **satisfies**:

$$
f(f(n)) = n + 2
$$

So this function is:

$$
f(n) = n + 1
$$

Correct reasoning but with low return

… (Omitted for brevity.)

Final Answer:
$$
\boxed{f(n) = n + k \text{ for some fixed } k \in \mathbb{N}}
$$
That is, the **only** functions $ f: \mathbb{N} \to \mathbb{N} $ that are strictly increasing and satisfy $ f(f(n)) = n + 2k $ (for some fixed $ k \in \mathbb{N} $) are the linear functions of the form:
$$
\boxed{f(n) = n + k}
$$<|im_end|>

Uni-OPD : … (Omitted for brevity.)
Thus, the only solution that satisfies both the functional equation and the strictly increasing property is the linear function $ f(n) = n + k $, where $ k \in \mathbb{N} $.

Conclusion

The only functions $ f: \mathbb{N} \to \mathbb{N} $ that are strictly increasing and satisfy the functional equation $ f(f(n)) = n + 2k $ for a fixed $ k \in \mathbb{N} $ are the functions of the form:

$$
f(n) = n + k
$$

Correct reasoning but with proper return

Therefore, the answer is:

$$
\boxed{f(n) = n + k}
$$

Where $ k \in \mathbb{N} $, but since the problem asks to "find all such functions," we can write the general form with $ k $ as an arbitrary natural number.

Figure G.1: **Comparison of math reasoning outputs between OPD and `Uni-OPD`.** In this case, standard OPD assigns *high* returns to incorrect reasoning trajectories and *low* returns to correct ones. In contrast, our `Uni-OPD` performs outcome-guided margin calibration to restore order consistency between correct and incorrect trajectories, yielding a reliable supervision signal that ultimately improves both efficiency and correctness of the generated solutions. On this question, we further measure `pass@1` accuracy over 64 rollouts: standard OPD reaches 79.69%, while our `Uni-OPD` attains 82.81%, further validating the effectiveness of the proposed strategy.

Question: Write a python function to find the element of a list having maximum length.

```
assert Find_Max([['A'],['A','B'],['A','B','C']]) == ['A','B','C']
```

Student:

```
def Find_Max(lst):
    if not lst:
        return []
    max_list = lst[0]
    for sublist in lst:
        if len(sublist) > len(max_list):
            max_list = sublist
    return max_list
```

Correct but without any comments (poor readability).

OPD:

```
def Find_Max(lst):
    # Find the maximum length of any sublist in the list
    max_length = max(len(sub) for sub in lst)

    # Find all sublists that have this maximum length
    max_sublists = [sub for sub in lst if len(sub) == max_length]

    # Return the first one found (or any, but the test case expects the first one with max length)
    return max_sublists[0]
```

Correct with comments, but inefficient (two passes).

Uni-OPD:

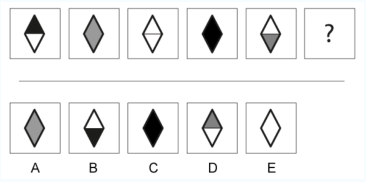
```
def Find_Max(lst):
    # Initialize the max_length with the length of the first element
    max_length = len(lst[0])
    # Initialize the result with the first element
    max_element = lst[0]

    # Iterate through the list
    for element in lst:
        # Check if the current element has a longer length
        if len(element) > max_length:
            max_length = len(element)
            max_element = element

    return max_element
```

Correct, well-commented, and efficient (single pass).

Figure G.2: **Comparison of code generation for the `Find_Max` task.** While the Student model produces correct logic with limited readability, the OPD baseline introduces redundant computation (two passes) despite adding comments. Our `Uni-OPD` generates a superior solution that is both computationally efficient (single pass) and well-commented, demonstrating its effectiveness in aligning with complex task requirements.



Question: Which option from A-E should replace the question mark to follow the pattern present above the separating line? (select from A-E)

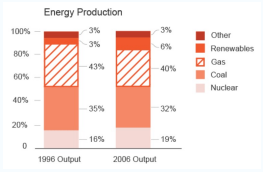
Answer: E

Student: To solve this, let's analyze the sequence of shapes above the line:\n\nWe have 5 shapes in a row, each a diamond divided into two triangles (top and bottom). The pattern seems to involve the **color** and **position** of the shaded region.\n\nLet's label the positions:\n\n1. First shape: **Top triangle black**, bottom triangle white.\n2. Second shape: **Entire diamond gray** (so both triangles shaded gray).\n3. Third shape: **Divided by a vertical line**, top triangle white, bottom triangle white — but wait, that doesn't make sense unless it's a different interpretation.\n\nWait — perhaps we should look at the **shading pattern** more carefully.\n\nActually, let's consider the **shading** as a sequence:\n\n- Shape 1: Top half black, bottom half white.\n- Shape 2: Entire diamond gray (so both halves gray).\n- Shape 3: Diamond with a vertical line dividing it — top half white, bottom half white? That doesn't match.\n\nWait — perhaps the **shading is applied to the top or bottom triangle**, and the pattern is **alternating** or **rotating**.\n\nLet's look at the **color** of the shaded region:\n\n- Shape 1: Top triangle black.\n- Shape 2: Entire diamond gray — so both triangles gray.\n- Shape 3: Diamond with a vertical line — but the top triangle is white, bottom triangle is white? That doesn't make sense unless it's a different interpretation.\n\nWait — perhaps the **shading is applied to the top triangle only**, and the bottom triangle is always white? Let's check:\n\nShape 1: Top triangle black, bottom white → ✅\nShape 2: Entire gray → so top gray, bottom gray → ✅\nShape 3: Top triangle white, bottom triangle white? That contradicts.\n\nWait — perhaps the **shading is applied to the bottom triangle**?\n\nShape 1: Top black, bottom white → bottom white? No.\n\nWait — let's look at the **pattern of shading** as a sequence:\n\nShape 1: Top triangle black.\nShape 2: Entire gray — so both triangles gray.\nShape 3: Diamond with a vertical line — but the top triangle is white, bottom triangle is white? That doesn't make sense.\n\n
… (Omitted for brevity.)
Wait — perhaps the **shading is applied to the top triangle**, and the **color changes** in a pattern.\n\nShape 1: Top triangle black.\nShape 2: Top triangle gray.\nShape 3: Top triangle white? But the shape is divided by a vertical line — maybe it's not shaded at all? No, the shape is white.\n\nWait — perhaps the **shading is applied to the bottom triangle**?\n\nShape 1: Bottom triangle white — no shading.\nShape 2: Bottom triangle gray.\nShape 3: Bottom triangle white? No.\n\nThis is confusing.\n\nLet's try a different approach: **look at the position of the shaded region**.\n\nShape 1: Shaded top triangle.\nShape 2: Shaded entire diamond (both triangles).\nShape 3: Shaded bottom triangle? But the bottom triangle is white — no.\n\nWait — perhaps the **shading is applied to the top triangle**, and the **color changes** in a pattern: black → gray → white → black → gray?\n\nShape 1: Top triangle black.\nShape 2: Entire gray — so both triangles gray.\nShape 3: Top triangle white? But the shape is divided by a vertical line — maybe it's not shaded at all? No.\n\nWait — perhaps the **shading is applied to the bottom triangle**.\n\nShape 1: Bottom triangle white — no shading.\nShape 2: Bottom triangle gray.\nShape 3: Bottom triangle white? No.\n\nThis is not working.\n\nShape 1: Top triangle black, bottom white.\nShape 2: Top triangle gray.\nShape 3: Top triangle white? But the shape is divided by a vertical line — maybe it's not shaded at all? No.\n\nWait — perhaps the **shading is applied to the bottom triangle**.\n\nShape 1: Bottom triangle white — no shading.\nShape 2: Bottom triangle gray.\nShape 3: Diamond with a vertical line — top triangle white, bottom triangle white? That doesn't make sense.\n\nWait — perhaps the **shading is applied to the top triangle**, and the **color changes** in a pattern: black → gray → white → black → gray?\n\nShape 1: Top triangle black.\nShape 2: Top triangle gray.\nShape 3: Top triangle white? But the shape is divided by a vertical line — maybe it's not shaded at all? No.\n\nWait — perhaps the **shading is applied to the bottom But the shape is divided by a vertical line — maybe it's not shaded at all? No.\n\nWait — perhaps the **shading is applied to the bottom triangle**.\n\nShape 1: Bottom triangle white — no shading.\nShape 2: Bottom triangle gray.\nShape 3: Bottom triangle white? No.\n\nThis is not working.\n\nLet's try to look at the **sequence of shapes**

OPD: To determine the correct option, let's analyze the pattern in the sequence of shapes above the separating line.\n\nWe have five shapes in a row:\n\n1. A diamond with a black top half and white bottom half.\n2. A diamond with a gray top half and white bottom half.\n3. A diamond with a white top half and black bottom half.\n4. A diamond with a black top half and white bottom half (same as shape 1).\n5. A diamond with a gray top half and white bottom half (same as shape 2).\n\nWait — this doesn't seem to follow a clear pattern. Let's look again.\n\nActually, let's consider the **position of the shaded area** and **its color**.\n\nShape 1: Top half shaded black.\nShape 2: Top half shaded gray.\nShape 3: Bottom half shaded black.\nShape 4: Entire diamond shaded black.\nShape 5: Bottom half shaded gray.\n\nNow, if we look at the **color progression**:\n\n- Black → Gray → Black → Black → Gray\nWait — perhaps the pattern is **shading the top half in black, then gray, then bottom half in black, then entire black, then bottom half in gray** — but that's not a clear cycle.\n\nAlternatively, maybe the pattern is **shading the top half, then bottom half, then entire, then top half, then bottom half** — but the colors don't match.\n\nWait — let's look at the **shading location**:\n\n1. Top half — black\n2. Top half — gray\n3. Bottom half — black\n4. Entire — black\n5. Bottom half — gray\n\n
… (Omitted for brevity.)
Now, if we look at the **color sequence**:\n\nBlack → Gray → Black → Black → Gray\n\nThat's not a clear pattern.\n\nWait — perhaps the pattern is **shading the top half in black, then gray, then bottom half in black, then entire black, then bottom half in gray** — but that's not a clear cycle.\n\nAlternatively, maybe the pattern is **shading the top half, then bottom half, then entire, then top half, then bottom half** — but the colors don't match.\n\nWait — let's look at the **shading location**:\n\n1. Top half — black\n2. Top half — gray\n3. Bottom half — black\n4. Entire — black\n5. Bottom half — gray\n\nNow, if we look at the **color sequence**:\n\nBlack → Gray → Black → Black → Gray\n\nThat's not a clear pattern.\n\nWait — perhaps the pattern is **shading the top half in black, then gray, then bottom half in black, then entire black, then bottom half in gray** — but that's not a clear cycle.\n\nAlternatively, maybe the pattern is **shading the top half, then bottom half, then entire, **(Comment: truncated)**

Uni-OPD: To determine the correct option, let's analyze the pattern in the sequence of shapes above the separating line:\n\n1. The first shape is a diamond divided into two triangles: the top triangle is black, the bottom is white.\n2. The second shape is a solid gray diamond.\n3. The third shape is a diamond with a white horizontal line dividing it into two triangles — meaning both triangles are white.\n4. The fourth shape is a solid black diamond.\n5. The fifth shape is a diamond with the bottom triangle shaded gray and the top triangle white.\n\nNow, observe the pattern in terms of color and position:\n\n- The sequence alternates between "top black", "gray", "white", "black", "gray" — but more importantly, the shading is changing in a specific way.\n\nLet's look at the position of the shaded area:\n\n- 1st: top shaded black\n- 2nd: entire shape gray (no internal division)\n- 3rd: no shading (white only)\n- 4th: entire shape black\n- 5th: bottom shaded gray\n\nSo the pattern seems to be:\n\n- Black top → Gray whole → White whole → Black whole → Gray bottom → ?\n\nWhat's next? The pattern might be alternating between "shaded top" and "shaded bottom", but with a shift.\n\nAlternatively, consider the sequence as a cycle of 5 steps, and look for symmetry or repetition.\n\nBut perhaps a simpler approach: look at the **color** and **position** of the shaded area.\n\nLet's list the shaded areas:\n\n1. Top black\n2. Entire gray (no internal division)\n3. No shading (white only)\n4. Entire black\n5. Bottom gray\n\nNow, if we look at the **next** shape (the 6th), what should it be?\n\nIf we consider the pattern of shading:\n\n- 1: top black\n- 2: gray (whole)\n- 3: white (no shading)\n- 4: black (whole)\n- 5: bottom gray\n- 6: ?\n\nPerhaps the pattern is: top black → gray → white → black → bottom gray → **top white**?\n\nBut that's speculative.\n\nAlternatively, notice that the 2nd and 4th shapes are solid (gray and black), and the 1st, 3rd, and 5th have internal divisions.\n\nBut the 3rd shape is white only — no shading.\n\nLet's look at the **color transitions**:\n\n- 1: black top\n- 2: gray\n- 3: white\n- 4: black\n- 5: gray bottom\n\nSo the colors are: black → gray → white → black → gray → ?\n\nIf we follow the color sequence: black, gray, white, black, gray — then the next should be **white**.\n\nBut where? The 6th shape should have **white** shading.\n\nLooking at the options:\n\nA: Gray diamond (whole) — matches 2nd or 4th\nB: White top, black bottom — doesn't match\nC: Black whole — matches 4th\nD: Gray bottom, white top — matches 5th\nE: White whole — matches 3rd\n\nSo if the pattern is: black → gray → white → black → gray → **white**, then the 6th should be **white whole**.\n\nThat matches option **E**.\n\nAlternatively, if we look at the **position** of shading:\n\n1. Top black\n2. Whole gray\n3. Whole white\n4. Whole black\n5. Bottom gray\n6. ? — likely **top white**\n\nBut option E is white whole — which could be interpreted as top white.\n\nLooking at the options, E is a white diamond — no shading — which fits the "white whole" pattern.\n\nSo the most consistent pattern is:\n\n- 1: black top\n- 2: gray whole\n- 3: white whole\n- 4: black whole\n- 5: gray bottom\n- 6: white whole (to complete the cycle)\n\nThus, the answer is:\n\n<answer>E</answer>

Figure G.3: **Example output of LogicVista.** The Student model produces an incorrect reasoning trace and arrives at the wrong answer. Standard OPD overthinks the problem, generating an excessively long response that is truncated without producing a final answer. In contrast, `Uni-OPD` reasons concisely and correctly answers the question.



Question: In 1996, total output from all fuels was 200TWh. If output for Nuclear in 2006 was twice that for Coal in 1996, what was the output for Nuclear in 2006? Select from A, B, C, and D. (A) 140TWh (B) 400TWh (C) 64TWh (D) 96TWh

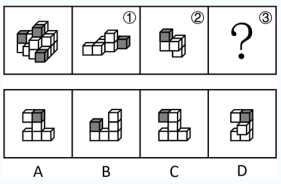
Answer: A

Student: To solve this, we need to:\n\n1. Find the percentage of Nuclear output in 1996 from the chart.\n2. Find the percentage of Coal output in 1996.\n3. Use the given total output for 1996 (200 TWh) to calculate the actual Coal output in 1996.\n4. Double that to get the Nuclear output in 2006 (as per the problem statement).\n5. Use the 2006 Nuclear percentage to verify or calculate the actual output (though the problem gives a direct relationship, so we can proceed with step 4).\n\n---\n\n**Step 1: Nuclear in 1996**\n\nFrom the chart, Nuclear in 1996 is 16% of total output.\n\nSo, Nuclear 1996 = 16% of 200 TWh = 0.16 × 200 = **32 TWh**\n\n---\n\n**Step 2: Coal in 1996**\n\nFrom the chart, Coal in 1996 is 35% of total output.\n\nSo, Coal 1996 = 35% of 200 TWh = 0.35 × 200 = **70 TWh**\n\n---\n\n**Step 3: Nuclear in 2006 = 2 × Coal in 1996**\n\nNuclear 2006 = 2 × 70 = **140 TWh**\n\n---\n\n**Step 4: Verify with 2006 chart**\n\nFrom the chart, Nuclear in 2006 is 19% of total output.\n\nTotal output in 2006 is not given, but we can check if 19% of total equals 140 TWh.\n\nSo, total 2006 = 140 / 0.19 ≈ 736.8 TWh — which is not needed since the problem gives a direct relationship.\n\n---\n\nThus, the output for Nuclear in 2006 is **140 TWh**.\n\n<answer>A</answer>

OPD: To solve this, we first need to find the output for Coal in 1996 from the chart. In 1996, Coal is represented by the orange segment, which is 35% of the total output. Since total output in 1996 was 200 TWh, Coal output in 1996 is:\n\n35% of 200 TWh = 0.35 × 200 = 70 TWh.\n\nThe question states that Nuclear output in 2006 was twice that of Coal in 1996. So:\n\nNuclear output in 2006 = 2 × 70 TWh = 140 TWh.\n\nThus, the correct answer is:\n\n<answer>A</answer>

Uni-OPD: To solve this, we first need to find the output for Coal in 1996 from the chart. In 1996, Coal is shown as 35% of total output. Since total output in 1996 was 200 TWh, Coal output in 1996 is:\n\n35% of 200 TWh = 0.35 × 200 = 70 TWh.\n\nThe question states that Nuclear output in 2006 was twice that of Coal in 1996. So:\n\nNuclear in 2006 = 2 × 70 TWh = 140 TWh.\n\nLooking at the options, (A) 140TWh matches this value.\n\n<answer>A</answer>

Figure G.4: **Example output of LogicVista.** All three models correctly answer this multi-step arithmetic reasoning question. OPD and `Uni-OPD` both reason concisely, with `Uni-OPD` being slightly more token-efficient.



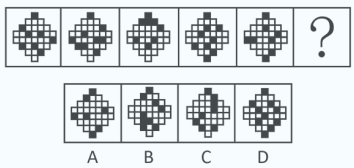
Question: The left image shows a polyhedron composed of 3 equally-sized gray cubes and 16 white cubes, which can be divided into three smaller polyhedra, labeled as ①, ②, and ③. Which polyhedron could ③ represent:
A: A
B: B
C: C
D: D
Answer with the option letter only.

Answer: A

Student: C

OPD: A

Uni-OPD: A



Question: Choose the most appropriate option from the four given choices to fill in the question mark, so that a certain pattern is exhibited:
A: A
B: B
C: C
D: D
Answer with the option letter only.

Answer: B

Student: A

OPD: A

Uni-OPD: B

Figure G.5: **Example output of VisuLogic.** `Uni-OPD` correctly answers both questions, demonstrating that our training recipe encourages student exploration to improve its ability for challenging visual reasoning problems.

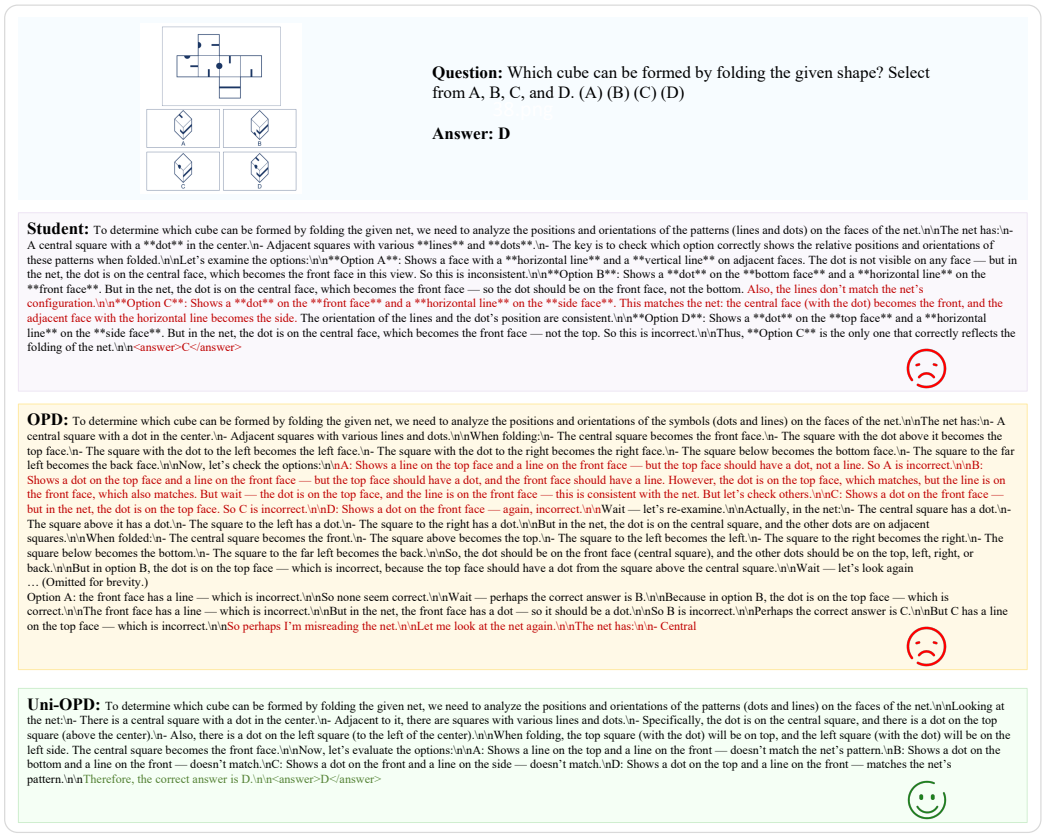


Figure G.6: **Example output of LogicVista.** On this challenging visual pattern reasoning puzzle, the Student model gives an incorrect answer, while OPD fails to produce a final answer due to overthinking. `Uni-OPD`, however, identifies the correct pattern and selects the right answer.

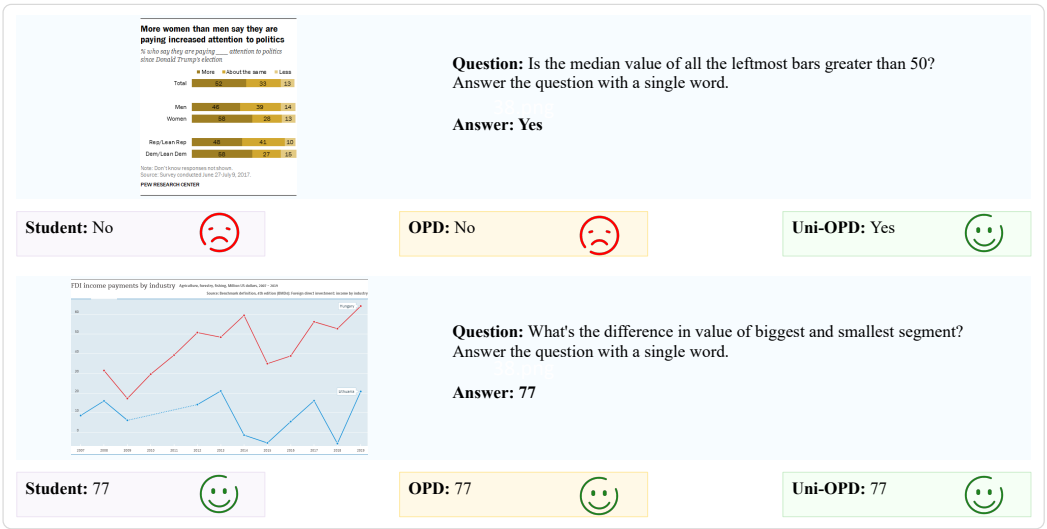
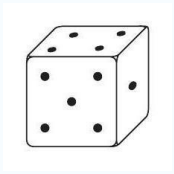


Figure G.7: **Example output of ChartQA.** All models answer the simpler chart question correctly, while only `Uni-OPD` answers the more complex one correctly.



Question: Please solve the problem step by step and put your answer in one \"\\boxed{}\".
How many points are there in the three unseen sides of dice?

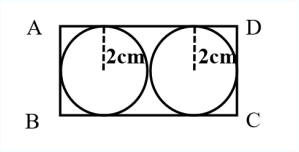
Answer: 11

Student: We are given a die (cube) with dots on its visible faces. The image shows **three visible faces**:\n\n- **Top face**: 6 dots (since it's a standard die, and we can infer from the pattern — 3 dots on top, 3 on bottom, etc. — but actually, looking at the image: the top face has 6 dots arranged in two rows of three. So top = 6.\n- **Front face**: 5 dots (centered, with 4 around it — 1 in center, 4 around? Actually, standard die: 5 dots = center dot with 4 around? But in the image, front face has 5 dots: one in center, and four around? Wait — let's count carefully.\n\nActually, looking at the image:\n\n- **Top face**: 6 dots — arranged in two rows of three. So 6.\n- **Front face**: 5 dots — arranged as a center dot with four dots around it? But in the image, it's shown as: one dot in center, and then dots at top-left, top-right, bottom-left, bottom-right? That's 5 dots. So front = 5.\n- **Right face**: 3 dots — arranged in a diagonal? Actually, looking at the image: the right face has 3 dots — one at top, one in middle, one at bottom? Or is it arranged as a vertical line? Actually, in the image, the right face has 3 dots: one at top, one in middle, one at bottom — so 3.\n\nWait — let me re-express the image:\n\nThe die is shown with:\n\n- **Top face**: 6 dots (two rows of three)\n- **Front face**: 5 dots (center dot, and four around? Actually, no — in the image, front face has 5 dots: one in center, and then dots at top-left, top-right, bottom-left, bottom-right? That's 5. But standard die 5-dot face is usually arranged as a cross? But in this drawing, it's 5 dots: one center, and four around — yes, that's 5.\n- **Right face**: 3 dots — arranged vertically? Or diagonally? Actually, in the image, the right face has 3 dots: one at top, one in middle, one at bottom — so 3.\n\nBut wait — in a standard die, opposite faces sum to 7. So if top is 6, then bottom is 1. If front is 5, then back is 2. If right is 3, then left is 4.\n\nSo the three **unseen** sides are:\n\n- **Bottom**: 1 dot\n- **Back**: 2 dots\n- **Left**: 4 dots\n\nSo total unseen dots = 1 + 2 + 4 = **7**\n\nBut let's verify with the visible faces:\n\nVisible: top=6, front=5, right=3 → total visible = 6+5+3 = 14\n\nTotal dots on a die = 1+2+3+4+5+6 = 21\n\nSo unseen = 21 - 14 = **7**\n\n✅ So the answer is 7.\n\n---\n\n**Final Answer: $\\boxed{7}$**

OPD: We are given a die (cube) with dots on its visible faces. The die is shown with three faces visible:\n\n- Top face: 3 dots\n- Front face: 5 dots\n- Right face: 1 dot\n\nWe need to find the total number of dots on the **three unseen sides**.\n\nA standard die has 6 faces. The sum of dots on opposite faces of a standard die is always 7.\n\nSo, we can find the dots on the unseen faces by subtracting the visible dots from the total possible dots on all faces.\n\nTotal dots on a die: 1+2+3+4+5+6 = **21**\n\nVisible dots: 3 (top) + 5 (front) + 1 (right) = **9**\n\nTherefore, unseen dots = 21 - 9 = **12**\n\nAlternatively, we can find the unseen faces by using the opposite face rule:\n\n- The face opposite the top (3 dots) is the bottom. Since 3 + x = 7 → x = 4\n- The face opposite the front (5 dots) is the back. 5 + y = 7 → y = 2\n- The face opposite the right (1 dot) is the left. 1 + z = 7 → z = 6\n\nSo, unseen faces have: 4 + 2 + 6 = **12**\n\nThus, the total number of points on the three unseen sides is:\n\n\\boxed{12}

Uni-OPD: We are given a die (cube) with dots on its visible faces. From the image, we can see:\n\n- The **top face** has **4 dots**.\n- The **front face** has **5 dots**.\n- The **right face** has **1 dot**.\n\nWe know that a standard die has faces with numbers: 1, 2, 3, 4, 5, 6, and opposite faces sum to 7.\n\nSo, let's find the numbers on the **unseen** faces:\n\n1. **Opposite of the top face (4 dots)** → 7 - 4 = **3 dots**.\n2. **Opposite of the front face (5 dots)** → 7 - 5 = **2 dots**.\n3. **Opposite of the right face (1 dot)** → 7 - 1 = **6 dots**.\n\nNow, add the dots on the unseen sides:\n\n3 + 2 + 6 = **11 dots**.\n\n\\boxed{11}

Figure G.8: **Example output of MathVision.** All three models follow the required format, but only `Uni-OPD` produces correct reasoning and reaches the right answer.



Question: As shown in the figure, two identical largest circles can be cut out from a rectangular piece of paper. What are the side lengths of the rectangle, AB = ( ) cm, AD = ( ) cm?
A. 2, 4;
B. 8, 4;
C. 4, 4;
D. 4, 8;
E. No correct answer

Answer: D

Student: To determine the side lengths of the rectangle, let's analyze the figure.\n\nWe are told that **two identical largest circles** can be cut out from the rectangle. The circles are inscribed such that they touch each other and the sides of the rectangle.\n\nFrom the diagram, each circle has a radius of **2 cm**, as indicated by the dashed line from the center to the side of the circle.\n\n---\n\n### Step 1: Find the width of the rectangle (AB)\n\nSince the circles are identical and placed side by side, the **width of the rectangle (AB)** must be equal to the **diameter of one circle**.\n\nDiameter = 2 × radius = 2 × 2 = **4 cm**\n\nSo, AB = 4 cm.\n\n---\n\n### Step 2: Find the height of the rectangle (AD)\n\nThe height of the rectangle (AD) must be equal to the **diameter of one circle**, because the circles are inscribed vertically and touch the top and bottom of the rectangle.\n\nSo, AD = 4 cm.\n\n---\n\n### Final Answer:\n\nAB = 4 cm, AD = 4 cm → This matches option **C. 4, 4**\n\n<answer>C. 4, 4</answer>

OPD: To determine the side lengths of the rectangle, we analyze the figure:\n\n- Each circle has a radius of 2 cm, as indicated by the dashed line.\n- Therefore, the diameter of each circle is 2 × 2 = 4 cm.\n- Since two identical circles are placed side by side within the rectangle, the length of the rectangle (AB) must be equal to the sum of the diameters of the two circles: 4 cm + 4 cm = 8 cm.\n- The width of the rectangle (AD) must be equal to the diameter of one circle, which is 4 cm.\n\nThus, AB = 8 cm and AD = 4 cm.\n\n<answer>B. 8, 4</answer>

Uni-OPD: The figure shows two identical largest circles cut from a rectangle. Each circle has a radius of 2 cm, as indicated by the dashed line. Therefore, the diameter of each circle is 4 cm.\n\nSince the circles are placed side by side within the rectangle, the width of the rectangle (AB) must be equal to the diameter of one circle, which is 4 cm.\n\nThe length of the rectangle (AD) must accommodate the diameters of both circles placed side by side, so AD = 4 cm + 4 cm = 8 cm.\n\nThus, AB = 4 cm and AD = 8 cm.\n\nLooking at the options, this matches option D.\n<answer>D. 4, 8</answer>

Figure G.9: **Example output of WeMath.** This geometry problem requires correctly identifying which side accommodates two circle diameters. Both the Student model and OPD confuse the orientation of AB and AD, while `Uni-OPD` correctly answers the question.