MAD-OPD: BREAKING THE CEILING IN ON-POLICY DISTILLATION VIA MULTI-AGENT DEBATE

Anonymous authors
Paper under double-blind review

ABSTRACT

On-policy distillation (OPD) trains a student on its own trajectories with token-level teacher supervision, but existing methods face a *single-teacher capability ceiling*: when the teacher and student make the same mistake, the teacher can reinforce it. This problem is sharper in agentic tasks, where per-step errors compound across long trajectories. We introduce MAD-OPD (Multi-Agent Debate-driven On-Policy Distillation), which uses debate among teachers to improve supervision at states visited by the student. The teachers use the debate transcript to evaluate the student's tokens and report confidence scores that determine their loss weights. For agentic tool use, we introduce On-Policy Agentic Distillation (OPAD), which performs step-wise OPD by sampling each tool action from the student and applying token-level supervision at that decision state. Our analysis of gradients and teacher aggregation motivates Jensen–Shannon divergence for agentic tasks and reverse Kullback–Leibler divergence for code generation. The divergence ablation supports these choices in the tested configuration. Across six teacher–student configurations spanning Qwen3 and Qwen3.5, 1.7B–14B students, and 8B–32B teachers on five agentic and code benchmarks, MAD-OPD ranks first in overall average. On Qwen3 14B+8B$\rightarrow$4B, it improves the agentic average by 4.39 points and the code average by 4.24 points over the stronger single-teacher OPD.

1 INTRODUCTION

On-policy distillation (OPD) has become a standard post-training method for modern LLMs (Lu & Thinking Machines Lab, 2025). It is used for strong-to-weak distillation in Qwen3 (Yang et al., 2025) and multi-teacher post-training in DeepSeek-V4 (DeepSeek-AI et al., 2026). Compared with outcome-reward reinforcement learning and off-policy sequence-level distillation (Kim & Rush, 2016), OPD provides dense teacher feedback at prefixes sampled from the student itself. Recent work adds privileged information (Zhao et al., 2026), reward extrapolation (Yang et al., 2026b), verifiable rewards (Yang et al., 2026a), and context internalization (Ye et al., 2026); Song & Zheng (2026) provide a broader survey.

Most OPD methods use a single teacher to supervise student-generated trajectories. When the teacher and student share an error, supervision can reinforce it, and even a stronger teacher may fail to improve the student (Li et al., 2026). Existing studies also center on reasoning benchmarks, especially mathematics (Song & Zheng, 2026; Li et al., 2026; Fu et al., 2026; Yang et al., 2026a), while multi-step agentic tasks (Xi et al., 2025) remain underexplored. In these tasks, an incorrect tool choice or argument can corrupt the context used by later decisions, making reliable supervision at each tool-decision step especially important. Divergence selection presents a further challenge: the divergence directly shapes the dense token-level objective (Yang et al., 2026b), but current methods generally treat it as a task-agnostic hyperparameter (Jang et al., 2026; Jin et al., 2026; Fu et al., 2026). Different tasks need different behavior when teacher and student distributions disagree, so a single default divergence is unlikely to fit both agentic tool use and code generation.

Multi-teacher distillation (Chen et al., 2024b) and multi-agent systems (Chen et al., 2025) show that combining models can improve reasoning and retrieval over single-model baselines. Multi-agent debate (MAD) lets agents refine their answers through iterative argumentation (Du et al., 2024), while confidence-weighted consensus can match or exceed the strongest individual model (Chen et al., 2024a). Most prior multi-teacher distillation methods, however, train on fixed teacher data rather than

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

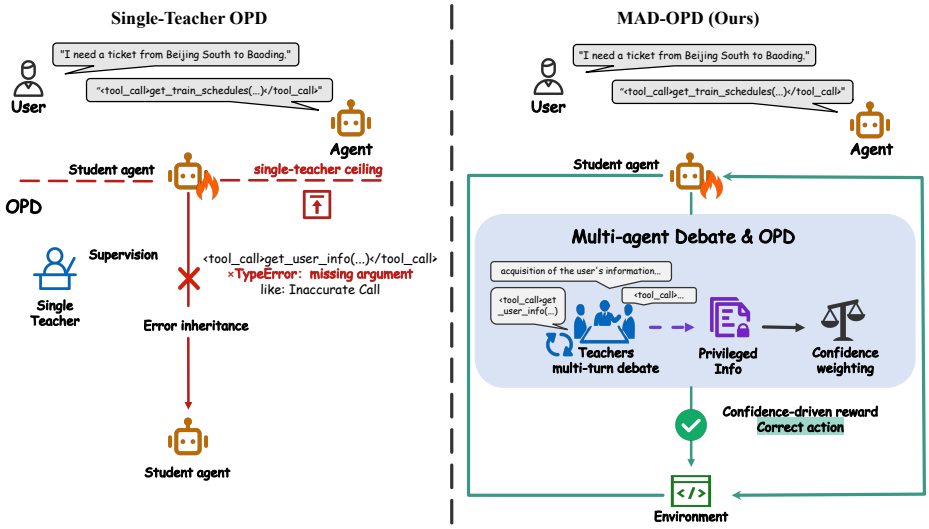


Figure 1: **Left:** a single teacher shares the student's tool-use error and reinforces it when supervising the student's sampled trajectory, exposing a *single-teacher capability ceiling*. **Right:** MAD-OPD's multi-round debate corrects each teacher's blind spots, producing supervision that outperforms individual teachers. App. E gives the full trajectory.

distributions computed on the student's current prefixes. Bringing debate into the on-policy loop allows teachers to inspect the same state the student actually visits, challenge one another's responses, and produce a stronger supervision signal for that state.

We propose MAD-OPD (Figure 1), the first framework to use multi-agent debate as token-level on-policy supervision. At each decision point, multiple teachers debate over the student's current state. The debate transcript provides privileged information to the teachers during force-decoding and remains hidden from the student. Post-debate confidence determines each teacher's contribution to the token-level loss. OPAD performs this distillation step at each tool decision, supervising student-sampled actions as the interaction proceeds. We analyze two properties of the distillation objective: JSD compares teacher and student distributions through a shared mixture, while confidence-weighted reverse KL induces a geometric pool of teacher distributions. These properties motivate our divergence choices for agentic tasks and code generation, respectively (Section 3.3). Section 5.4 compares their task-level performance.

Our contributions are summarized as follows:

- **Divergence analysis and selection.** We analyze how distribution mismatch affects OPD gradients and how reverse KL combines teacher distributions. The analysis motivates JSD for agentic tasks and reverse KL for code generation, with both choices supported by our divergence ablation.
- **Multi-Agent Debate-driven OPD framework.** We introduce MAD-OPD, the first framework to use multi-agent debate as token-level on-policy supervision. It supervises the student using teacher distributions conditioned on the debate transcript.
- **OPAD for agentic tasks.** We propose *On-Policy Agentic Distillation* (OPAD), which samples and supervises student actions at individual tool-decision steps, extending OPD to multi-step tool use.

2 Related Work

On-Policy Distillation. Knowledge distillation (Hinton et al., 2015) transfers a teacher's predictions to a smaller student, while sequence-level distillation trains on teacher-generated outputs (Kim & Rush, 2016). MiniLLM uses reverse KL to distill from student-generated sequences (Gu et al., 2024), and Agarwal et al. (2024) formulate OPD as dense token-level supervision on student trajectories. OPD has since been adopted in frontier-model post-training (Yang et al., 2025; DeepSeek-AI et al., 2026) and surveyed by Song & Zheng (2026). Recent work studies its failure modes (Li et al., 2026;

Fu et al., 2026), stabilizes the reverse-KL objective (Jang et al., 2026; Jin et al., 2026), and introduces privileged information available only to the teacher (Penaloza et al., 2026). Most OPD studies use one teacher and a fixed divergence.

Multi-Agent Systems. Multi-agent systems (MAS) coordinate specialized agents, and Chain-of-Agents (Li et al., 2025) distills an MAS pipeline into a single foundation model. Multi-agent debate (MAD) improves reasoning by letting agents revise their answers after seeing peer responses (Du et al., 2024; Liang et al., 2024; Chen et al., 2024a). The inference-time MAD methods above do not use debate distributions as token-level on-policy supervision. MAD-OPD brings this signal into training.

Multi-Teacher Distillation. Multi-teacher distillation pools signals from several teachers. Sequence-level distillation trains on teacher-generated outputs (Kim & Rush, 2016), while AMTSS combines teacher importance weights with adaptive teacher selection (Chen et al., 2023). Recent methods distill multi-agent interaction graphs, debate traces, or teacher-generated reasoning (Chen et al., 2024b; Aluru et al., 2025; Zhou et al., 2025; Shen et al., 2025). Their objectives use fixed interaction traces or teacher-generated data rather than teacher distributions evaluated on the student's current token prefixes. MAD-OPD brings inter-teacher debate into the on-policy supervision loop.

3 Preliminaries and Divergence Analysis

3.1 On-Policy Distillation

Training objective. Given a teacher π^* and a student π_θ with parameters θ, OPD minimizes a divergence along the student's own trajectories (Agarwal et al., 2024; Song & Zheng, 2026). For input x from data distribution $\mathcal{D}$ and $\hat{y} \sim \pi_\theta(\cdot|x)$:

$$\mathcal{L}_{\text{OPD}}(\theta) = \mathbb{E}_{x,\,\hat{y}\sim\pi_\theta}\left[\frac{1}{|\hat{y}|}\sum_{t=1}^{|\hat{y}|} D\big(\pi^*(\cdot|x,c,\hat{y}_{<t}) \;\big\|\; \pi_\theta(\cdot|x,\hat{y}_{<t})\big)\right], \tag{1}$$

where D is a divergence and c is additional context available only to the teacher (Zhao et al., 2026). In our method, c is the debate transcript. During optimization, we hold the sampled trajectory and teacher distributions fixed and differentiate the divergence through the student (Lu & Thinking Machines Lab, 2025; Yang et al., 2026b; Hübotter et al., 2026). App. D.1 gives the gradient estimator.

Notation. For brevity, when the position t is clear from context we write $p \triangleq \pi^*(\cdot|x,c,\hat{y}_{<t})$ for the privileged teacher distribution and $q \triangleq \pi_\theta(\cdot|x,\hat{y}_{<t})$ for the student distribution over the vocabulary $\mathcal{V}$. The student's pre-softmax logit at coordinate i is z_i, so $q(v) = \mathrm{softmax}(z)_v$. All logarithms are natural (base e).

Teacher–student distribution mismatch. The teacher conditions on c, while the student uses the task context and its own prefix. At a state visited by the student, the additional context can change which tokens the teacher considers plausible. A token sampled with appreciable student probability may then receive very little teacher probability. With multiple teachers, their conditional distributions can also favor different valid continuations. We examine how divergences respond to the first mismatch and aggregate the latter distributions; these are the two properties underlying our task-dependent choices.

3.2 Theoretical Properties of Divergences

Three divergences are standard candidates for D in OPD:

$$\text{Forward KL:}\quad D^{\rightarrow}_{\mathrm{KL}}(p\|q) = \textstyle\sum_v p(v)\log\frac{p(v)}{q(v)}, \tag{2}$$

$$\text{Reverse KL:}\quad D^{\leftarrow}_{\mathrm{KL}}(p\|q) \triangleq D_{\mathrm{KL}}(q\|p) = \textstyle\sum_v q(v)\log\frac{q(v)}{p(v)}, \tag{3}$$

$$\text{JSD:}\quad \mathrm{JSD}_\beta(p\|q) = \beta D_{\mathrm{KL}}(p\|m) + (1-\beta)D_{\mathrm{KL}}(q\|m), \quad m = \beta p + (1-\beta)q. \tag{4}$$

We compare the gradients of these divergences and how they combine teacher distributions. For JSD, we use $\beta = 0.5$, whose maximum value is $\log 2$ (Lemma 1.1).

JSD under distribution mismatch. JSD compares both distributions with the mixture $m = \beta p + (1-\beta)q$. Since $m(v) \geq \beta p(v)$ and $m(v) \geq (1-\beta)q(v)$, the mixture assigns positive probability to every token supported by either distribution. This reference remains defined when the teacher assigns zero probability to a token supported by the student. Both terms in JSD compare against the same mixture, whose dependence on q must be included when differentiating. The resulting gradient is

$$\frac{\partial \mathrm{JSD}_\beta(p\|q)}{\partial z_i} = (1-\beta)q(i)\left[\log\frac{q(i)}{m(i)} - \mathbb{E}_{v\sim q}\log\frac{q(v)}{m(v)}\right], \tag{5}$$

which depends on $\log(q/m)$. The mixture bounds the logit gradient even when $p(i)$ approaches zero, as shown below.

Lemma 1 (Bounds on JSD and its logit gradient)**.** *For any distributions p, q over $\mathcal{V}$ and $\beta \in (0,1)$:*

1. ***Loss bound (Lin, 1991):*** $\mathrm{JSD}_\beta(p\|q) \in [0,\ H(\beta)]$*, where* $H(\beta) = -\beta\log\beta - (1-\beta)\log(1-\beta)$*. For* β=0.5*:* $\mathrm{JSD}_{0.5} \in [0,\ \log 2]$.
2. ***Logit-gradient bound:*** *with p fixed and* $q = \mathrm{softmax}(z)$*,* $\|\nabla_z \mathrm{JSD}_\beta(p\|q)\|_\infty \leq 2$ *uniformly for every* $\beta \in (0,1)$*, regardless of support overlap (App. D.2).*

Proof in App. D.2.

Multi-teacher reverse KL. For a single teacher and unrestricted q, reverse KL is minimized at $q = p$. With multiple teachers, the objective sums their confidence-weighted reverse-KL losses. Teacher log-probabilities then combine linearly, giving a normalized weighted geometric mean as the effective target (Proposition 2). A token assigned low probability by one teacher is penalized according to that teacher's weight. This connects the distillation objective to the teachers' agreement after debate.

3.3 Divergence Analysis and Task-Dependent Choices

We analyze how divergences respond to teacher–student support mismatch and combine supervision from multiple teachers. These properties guide the divergence choices used in MAD-OPD.

Agentic tasks: gradients under distribution mismatch. A multi-step agentic trajectory $\tau = (a_1, o_1, \ldots, a_M, o_M)$ conditions later decisions on earlier actions and observations. A tool-use error can therefore affect subsequent steps. When the privileged teacher assigns near-zero probability to a student-supported token, the divergence determines the strength of the local correction. The OPD update differentiates the divergence at each sampled prefix (App. D.1).

Proposition 1 (Logit-gradient bounds under distribution mismatch)**.** *Under the dense OPD gradient estimator, the per-token logit gradient satisfies:*

1. ***JSD:*** *the mixture keeps both teacher- and student-supported tokens in the reference support, and* $\|\nabla_{z_t}\mathrm{JSD}_\beta(p\|q)\|_\infty \leq 2$ *uniformly (Lemma 1.2). Under the global-token mean or OPAD's sum of per-step means,* $\|\nabla_z \mathcal{L}_{\mathrm{JSD}}(\tau)\|_\infty \leq 2$*, independent of trajectory length and support overlap.*
2. ***Reverse KL:*** $\|\nabla_{z_t} D^{\leftarrow}_{\mathrm{KL}}(p\|q)\|_\infty$ *has no uniform bound over positive teacher distributions. For fixed q with* $q(i) \in (0,1)$*, it diverges as* $p(i) \to 0$ *while all other teacher probabilities stay bounded away from zero. Both loss reductions retain this unboundedness for any fixed sampled trajectory.*
3. ***Forward KL:*** $\|\nabla_{z_t} D^{\rightarrow}_{\mathrm{KL}}(p\|q)\|_\infty = \|q-p\|_\infty \leq 1$ *is bounded. Its relation to the empirical ranking is discussed in App. D.5.*

Proof sketch. Part 1 follows from Lemma 1.2 and the block structure of per-position logits. Part 2 uses the centered reverse-KL gradient, whose coefficient on $\log(1/p(i))$ is $q(i)(1-q(i)) > 0$. Part 3 follows from $|q(i) - p(i)| \leq 1$. These are logit-space guarantees; parameter gradients also depend on network Jacobians. Full derivations are in App. D.3.

Code generation: combining teacher distributions. A code problem can have several valid solutions, each requiring a consistent sequence of implementation choices. With reverse KL, the per-teacher objective in Eq. 8 combines the post-debate teachers in log-probability space.

Proposition 2 (Geometric pooling under reverse KL)**.** *At a student-visited prefix, let $p_1, \dots, p_K$ be positive teacher token distributions and $w_k \geq 0$ with $\sum_k w_k = 1$. For the normalized geometric pool $\bar{p}_{\mathrm{geo}}(v) = \prod_k p_k(v)^{w_k} / Z_{\mathrm{geo}}$, where $Z_{\mathrm{geo}} = \sum_u \prod_k p_k(u)^{w_k}$,*

$$\sum_{k=1}^{K} w_k D_{\mathrm{KL}}(q \| p_k) = D_{\mathrm{KL}}(q \| \bar{p}_{\mathrm{geo}}) - \log Z_{\mathrm{geo}}, \tag{6}$$

so the objective is minimized at $q = \bar{p}_{\mathrm{geo}}$. Its log-odds are the confidence-weighted teacher log-odds, strengthening token choices supported across the post-debate teachers.

Derivation in App. D.4.

Divergence choices. These properties motivate JSD for tool use and reverse KL for code generation. In the 14B+8B→4B ablation with shared hyperparameters, JSD achieves the highest agentic average and reverse KL the highest code average (Table 7). For tool use, the mixture provides control of local corrections when privileged context makes the teacher and student disagree sharply. For code, geometric pooling combines the teachers' preferences at each prefix, motivating debate before aggregation so that teachers can reconcile their proposed solutions. These properties provide design grounds for the choices, while the task-level ranking comes from the ablation. We use these divergences throughout the experiments.

4 METHOD: MAD-OPD

MAD-OPD improves supervision on student rollouts by using multi-teacher debate as privileged context. In ordinary OPD, a teacher evaluates the tokens sampled by the student, but can reinforce an incorrect action when it makes the same mistake. Our intuition is that teachers can expose and correct complementary errors by comparing their answers at the same decision state. They then use the debate transcript when assigning token probabilities to the student's action, with their final confidence scores determining the loss weights. The student learns from these distributions using only its own context, transferring the benefit of debate into its policy. For tool use, OPAD couples this supervision with the student's interaction loop: each action changes the observed state, and teachers reassess the next decision in that updated context.

4.1 DEBATE AMONG TEACHERS

Given K teachers $\{T_1, \dots, T_K\}$ and the current state s_m, debate runs for R rounds. For single-turn tasks, s_m is the prompt x; for multi-step agentic tasks, it is the context $(x, \tau_{<m})$. At round 1, each teacher answers from s_m independently. At every subsequent round, each teacher revises its response h_r^k after reading all earlier responses:

$$h_r^k \sim p_{T_k}\big(\cdot \mid s_m,\ \{h_{r'}^j\}_{j=1,\dots,K,\ r'<r}\big), \quad k = 1, \dots, K, \ \ r = 1, \dots, R. \tag{7a}$$

Differences in model capacity and parameterization yield diverse initial responses. The subsequent exchange lets teachers challenge one another and catch errors that a single teacher may miss. The full debate history is $\mathcal{H}_m^R = \{h_r^k\}_{k,r}$.

We use $c_m = \mathcal{H}_m^R$ as the additional teacher context in Eq. 1. The student receives the task context and its own prefix; the debate transcript is available only to the teachers.

4.2 WEIGHTING TEACHERS BY CONFIDENCE

Each teacher ends its final-round response h_R^k with an integer confidence score $c_k \in \{0, \dots, 100\}$. We extract this score from the final line (App. F):

$$c_k = \mathrm{parse_confidence}(h_R^k). \tag{7b}$$

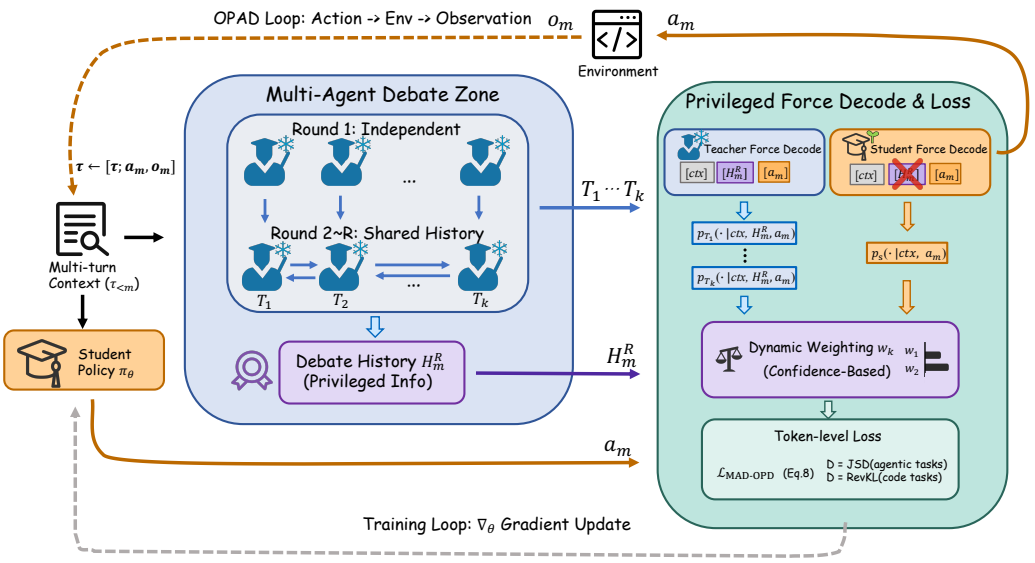


Figure 2: **The MAD-OPD pipeline.** At trajectory step m, the student π_θ samples an on-policy action a_m; K teachers debate for R rounds to produce a transcript $\mathcal{H}_m^R$ visible only to the teachers. Teachers then force-decode a_m and contribute to a confidence-weighted divergence D (JSD for agents, $D_{\text{KL}}^{\leftarrow}$ for code); gradients update only π_θ. The orange path marks the OPAD outer loop where the environment returns observation o_m for the next step.

We divide each score by 100 and apply softmax to the resulting values $\tilde{c}_k := c_k/100 \in [0, 1]$:

$$w_k = \frac{\exp(\tilde{c}_k/\tau_{\text{conf}})}{\sum_{j=1}^{K} \exp(\tilde{c}_j/\tau_{\text{conf}})}, \qquad \sum_{k=1}^{K} w_k = 1, \tag{7c}$$

with $\tau_{\text{conf}} = 1.0$. A higher reported confidence gives the teacher a larger loss weight.

4.3 Token-Level Distillation Objective

Each teacher predicts the next token at every prefix of the student's sample $\hat{y}$, using the current state s_m and debate transcript $\mathcal{H}_m^R$. Write $p_{k,t} = p_{T_k}(\cdot \mid s_m, \mathcal{H}_m^R, \hat{y}_{<t})$ for teacher k and $q_t = \pi_\theta(\cdot \mid s_m, \hat{y}_{<t})$ for the student. At this state, the loss is

$$\mathcal{L}_{\text{MAD-OPD}}(\theta; s_m) = \mathbb{E}_{\hat{y}\sim\pi_\theta(\cdot|s_m)}\left[\frac{1}{|\hat{y}|}\sum_{t=1}^{|\hat{y}|}\sum_{k=1}^{K} w_k D(p_{k,t}\|q_t)\right]. \tag{8}$$

Following Section 3.3, we set $D = \text{JSD}_\beta$ for agentic tasks and $D = D_{\text{KL}}^{\leftarrow}$ for code generation. Divergences span the full vocabulary; teacher logits are treated as fixed targets, so gradients flow only through the student. Because $w_k \geq 0$, $\sum_k w_k = 1$, and the confidence weights are detached, the weighted multi-teacher JSD loss preserves the coordinatewise bound in Proposition 1.

4.4 On-Policy Agentic Distillation (OPAD)

On-Policy Agentic Distillation (OPAD) applies OPD at each turn of the interaction. We instantiate OPAD with MAD-OPD; the same procedure also supports single-teacher OPD. The student generates an on-policy trajectory $\tau = (a_1, o_1, \ldots, a_M, o_M)$ of length M: at step m, with state $s_m = (x, \tau_{<m})$, it samples action $a_m \sim \pi_\theta(\cdot \mid s_m)$ and the environment returns $o_m \sim P^{\text{env}}(\cdot \mid s_m, a_m)$. At each step, teachers debate on s_m and force-decode a_m conditioned on the resulting transcript $\mathcal{H}_m^R$, giving the per-step loss:

$$\mathcal{L}_D^{\text{opad}}(s_m, a_m) = \frac{1}{|a_m|}\sum_{t=1}^{|a_m|}\sum_{k=1}^{K} w_k D\big(p_{T_k}(\cdot \mid s_m, \mathcal{H}_m^R, a_{m,<t}) \,\big\|\, \pi_\theta(\cdot \mid s_m, a_{m,<t})\big). \tag{9}$$

The total trajectory loss sums per-step contributions over the on-policy rollout:

$$\mathcal{L}_{\mathrm{OPAD}}(\theta) = \mathbb{E}_{x\sim\mathcal{D},\,\tau\sim\pi_\theta(\cdot|x)}\left[\sum_{m=1}^{M}\mathcal{L}_D^{\mathrm{opad}}(s_m, a_m)\right]. \tag{10}$$

Because debate occurs at every step and conditions on the observed history $\{o_1, \ldots, o_{m-1}\}$, supervision follows the trajectory generated by the student. Algorithm 1 in App. A gives the complete procedure.

5 Experiments

We evaluate MAD-OPD across six teacher–student configurations, two model families, and five benchmarks. Four research questions guide the analysis: **(RQ1)** Does debate break the single-teacher capability ceiling? **(RQ2)** How do gains vary with student size, teacher pair, and model family? **(RQ3)** How much do debate and confidence weighting each contribute? **(RQ4)** How does divergence choice affect agentic and code performance?

5.1 Experimental Setup

We evaluate MAD-OPD on Qwen3 (Yang et al., 2025) and Qwen3.5 (Qwen Team, 2026) across six teacher–student configurations. The Qwen3 teacher pairs are 14B+8B and 32B+30B-A3B; the latter uses `Qwen3-30B-A3B-Instruct-2507`. The Qwen3.5 pair is 27B+9B, and student sizes range from 1.7B to 14B. Agentic training uses approximately 16K ToolACE examples (Liu et al., 2025); code training uses 30K OpenThoughts3 problems (Guha et al., 2025).

The benchmarks instantiate the agentic/code split studied in Sec. 3.3. Agentic evaluation covers BFCL-v4 (Patil et al., 2025), τ^2-Bench (Barres et al., 2025), and VitaBench (He et al., 2025); code evaluation covers LiveCodeBench v6 (Jain et al., 2025) and MBPP+ (Austin et al., 2021; Liu et al., 2023). We focus on these two regimes because recent OPD work already studies mathematical reasoning extensively (Li et al., 2026; Fu et al., 2026; Yang et al., 2026a).

We compare four reference paradigms: the undistilled instruct model (**Base**); single-teacher OPD with the stronger teacher of each pair (Agarwal et al., 2024) (**OPD**); uniform multi-teacher OPD without debate (**MT-OPD**); and off-policy multi-teacher sequence distillation with a 7:3 strong-to-weak mix (**MT-SeqKD**) (Kim & Rush, 2016). These comparisons isolate how the teacher supervision is constructed within distillation.

All methods use the same student-facing input format, supervised-token mask, action format, parser, and inference budget. The main matrix uses non-thinking generation; App. C.2 reports native-thinking results. App. B gives the prompts, hyperparameters, and full protocol.

5.2 Main Results (RQ1)

Debate supervision breaks the single-teacher ceiling. MAD-OPD ranks first in overall average across all six configurations. In 14B+8B→4B, it improves Ag-Avg by 4.39 points and Co-Avg by 4.24 over single-teacher OPD. The 4B student also exceeds the 14B teacher on LCB-v6 by 4.30 pass@1 and 10.29 BoN@16 points (App. C.6), showing that the student can outperform the stronger teacher. MT-OPD trails OPD on code in five configurations, and MT-SeqKD remains below MAD-OPD in every overall comparison. These comparisons suggest that the benefit depends on how teachers construct supervision on the student's trajectory, beyond simply increasing the number of teachers.

Teacher exchange contributes beyond additional compute. Enabling peer-response visibility improves Overall by 3.11 points, and MAD-OPD exceeds OPD by 1.92 Ag-Avg points under equal training time (App. C.1). It also outperforms OPD on all five benchmarks with native thinking enabled (App. C.2).

Table 1: **Main results in percent.** Agentic tasks use $D{=}\mathrm{JSD}_\beta$, and code tasks use $D{=}D_{\mathrm{KL}}^{\leftarrow}$ as specified in Section 3.3. **Ag-Avg** and **Co-Avg** average the three agentic and two code benchmarks, respectively; **Overall** averages all five. Code pass@1 is averaged over 16 random seeds, with standard deviations in subscripts. **Bold**: best; underline: second best per configuration.

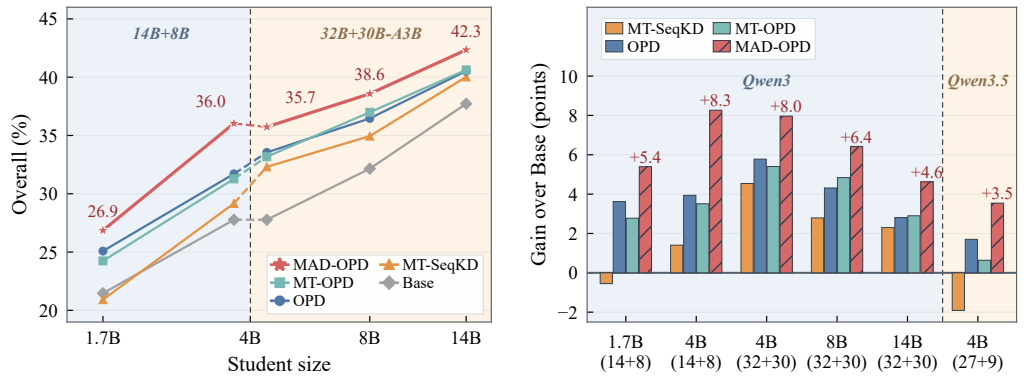
	Agentic Tasks ($D{=}$JSD)				Code Tasks ($D{=}D_{\mathrm{KL}}^{\leftarrow}$)			
Method	BFCL-v4	τ^2	Vita	Ag-Avg	LCB-v6	MBPP+	Co-Avg	**Overall**
Qwen3 14B+8B → 1.7B								
Base	19.65	13.47	7.50	13.54	$15.43_{\pm 0.61}$	$51.27_{\pm 0.63}$	33.35	21.46
MT-SeqKD	15.52	13.02	9.00	12.51	$15.82_{\pm 0.74}$	$51.22_{\pm 0.75}$	33.52	20.92
OPD	20.91	18.72	12.25	17.29	$19.26_{\pm 0.84}$	$54.28_{\pm 0.78}$	36.77	25.08
MT-OPD	20.96	18.07	11.25	16.76	$19.07_{\pm 0.73}$	$51.85_{\pm 1.14}$	35.46	24.24
MAD-OPD	**23.14**	**20.59**	**13.25**	**18.99**	**21.39**$_{\pm 1.21}$	**55.92**$_{\pm 1.03}$	**38.66**	**26.86**
Qwen3 14B+8B → 4B								
Base	23.34	18.00	14.50	18.61	$22.73_{\pm 0.65}$	$60.27_{\pm 0.35}$	41.50	27.77
MT-SeqKD	22.51	26.07	15.75	21.44	$23.18_{\pm 0.21}$	$58.36_{\pm 1.32}$	40.77	29.17
OPD	24.79	29.17	15.75	23.24	$25.00_{\pm 0.52}$	$63.82_{\pm 1.21}$	44.41	31.71
MT-OPD	25.65	29.06	17.75	24.15	$23.29_{\pm 0.41}$	$60.64_{\pm 0.72}$	41.97	31.28
MAD-OPD	**28.39**	**33.74**	**20.75**	**27.63**	**29.87**$_{\pm 0.56}$	**67.42**$_{\pm 0.92}$	**48.65**	**36.03**
Qwen3 32B+30B-A3B → 4B								
Base	23.34	18.00	14.50	18.61	$22.73_{\pm 0.65}$	$60.27_{\pm 0.35}$	41.50	27.77
MT-SeqKD	23.09	**35.60**	14.50	24.40	$25.06_{\pm 0.26}$	$63.29_{\pm 1.15}$	44.17	32.31
OPD	27.70	32.39	16.25	25.45	$26.96_{\pm 0.41}$	$64.43_{\pm 1.03}$	45.70	33.55
MT-OPD	24.95	34.60	17.25	25.60	$27.14_{\pm 0.43}$	$61.92_{\pm 0.85}$	44.53	33.17
MAD-OPD	**27.76**	34.62	**19.50**	**27.29**	**30.42**$_{\pm 0.43}$	**66.37**$_{\pm 0.95}$	**48.40**	**35.73**
Qwen3 32B+30B-A3B → 8B								
Base	28.94	30.08	17.25	25.42	$20.93_{\pm 0.76}$	$63.56_{\pm 0.61}$	42.25	32.15
MT-SeqKD	30.62	39.06	16.75	28.81	$21.93_{\pm 0.31}$	$66.36_{\pm 1.52}$	44.14	34.94
OPD	29.94	36.91	22.00	29.62	$26.83_{\pm 0.34}$	$66.64_{\pm 0.93}$	46.73	36.46
MT-OPD	29.96	38.73	22.50	30.40	$26.57_{\pm 0.38}$	$67.20_{\pm 1.31}$	46.89	36.99
MAD-OPD	**31.39**	**40.21**	**23.25**	**31.62**	**30.71**$_{\pm 0.74}$	**67.32**$_{\pm 1.12}$	**49.02**	**38.58**
Qwen3 32B+30B-A3B → 14B								
Base	32.46	33.70	21.75	29.30	$25.57_{\pm 0.87}$	$75.13_{\pm 0.78}$	50.35	37.72
MT-SeqKD	33.99	40.26	23.25	32.50	$27.25_{\pm 1.01}$	$75.36_{\pm 0.85}$	51.30	40.02
OPD	32.16	40.53	24.50	32.40	$28.73_{\pm 0.84}$	$76.73_{\pm 1.04}$	52.73	40.53
MT-OPD	30.52	42.42	**25.50**	32.81	$29.17_{\pm 0.84}$	$75.50_{\pm 0.62}$	52.34	40.62
MAD-OPD	**34.35**	**43.59**	25.00	**34.31**	**31.36**$_{\pm 1.02}$	**77.42**$_{\pm 1.31}$	**54.39**	**42.34**
Qwen3.5 27B+9B → 4B								
Base	35.76	42.25	9.75	29.25	$21.93_{\pm 1.02}$	$62.73_{\pm 0.98}$	42.33	34.48
MT-SeqKD	24.04	35.54	7.50	22.36	$29.06_{\pm 0.62}$	$66.73_{\pm 0.97}$	47.90	32.57
OPD	34.76	43.39	9.25	29.13	$28.32_{\pm 1.02}$	$65.21_{\pm 0.57}$	46.77	36.19
MT-OPD	33.98	39.53	10.50	28.00	$28.12_{\pm 0.42}$	$63.49_{\pm 0.94}$	45.80	35.12
MAD-OPD	**36.72**	**44.83**	**11.00**	**30.85**	**30.26**$_{\pm 0.46}$	**67.32**$_{\pm 1.15}$	**48.79**	**38.03**

5.3 Scaling Behavior (RQ2)

MAD-OPD generalizes across student sizes, teacher pairs, and model families. In Figure 3, the gain over Base with 14B+8B teachers grows from 5.39 points at 1.7B to 8.27 at 4B. For the 4B student, 14B+8B debate reaches 36.03 Overall, above both 32B single-teacher OPD (33.55) and 32B+30B-A3B debate (35.73). This is consistent with the importance of compatibility between teacher and student thinking patterns in OPD (Li et al., 2026). On Qwen3.5, MAD-OPD gains 3.54 points over Base versus at most 1.70 for the baselines. The scaling comparison suggests that a larger teacher alone is insufficient to predict distillation quality; a smaller pair that exchanges complementary information can provide more effective supervision.

5.4 Component and Divergence Analysis (RQ3, RQ4)

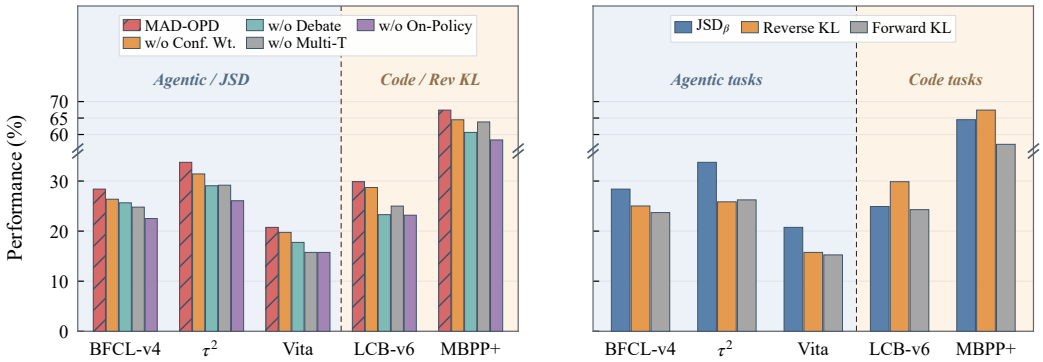
Debate and confidence weighting improve different parts of supervision. In Figure 4(a), uniform MT-OPD trails single-teacher OPD on code, whereas adding debate with uniform weights improves Co-Avg by 4.63 points. This supports reconciling teacher responses before aggregating their token distributions, especially when teachers propose different implementations. Confidence weighting



(a) Qwen3: teacher and student scaling

(b) Gains across all six configurations

Figure 3: **Scaling behavior. (a)** Overall score versus Qwen3 student size; dashed segments connect teacher pairs at 4B. **(b)** Gains over Base across six configurations. Background shading identifies teacher pairs in (a) and model families in (b).



(a) Component ablation

(b) Divergence selection

Figure 4: **Ablations** on 14B+8B→4B: **(a)** component ablation; **(b)** divergence selection.

then improves both task averages, suggesting that revising responses and adjusting their contributions are both useful (Table 6). The advantage over MT-SeqKD also supports the full framework's use of supervision at student-visited states, although that comparison changes both the sampling and supervision objectives. Two rounds outperform both independent responses and a third round (App. C.9), favoring a single exchange of teacher feedback in this setting.

Divergence choice depends on how teacher disagreement is handled. With shared hyperparameters, JSD leads on all three agentic benchmarks and reverse KL on both code benchmarks (Figure 4(b)). JSD's mixture bounds local logit gradients when privileged teacher context creates sharp disagreement with the student. Forward KL also has bounded gradients but performs worse, so boundedness alone does not explain the ranking. For code, reverse KL induces a geometric pool that favors token choices supported across teachers at the current prefix; debate can make these preferences more consistent before distillation. These distributional properties help interpret the observed task split and motivate our choices (Section 3.3); full scores appear in App. C.5.

6 Discussion and Conclusion

We introduced MAD-OPD, which turns multi-agent debate into token-level on-policy supervision, together with OPAD for step-wise agentic training. Our divergence analysis and ablation support using JSD for agentic tasks and reverse KL for code generation. Across six teacher–student configurations and five benchmarks, MAD-OPD ranks first in every overall comparison. On LCB-v6, its 4B student exceeds the 14B teacher by 4.30 pass@1 points and 10.29 BoN@16 points. The results show that multi-agent debate can provide stronger supervision than a single teacher while extending OPD to agentic tool use.

AI USE STATEMENT

We used generative AI only to edit and polish the paper's writing, including grammar, wording, concision, and readability. It did not contribute to the research ideas, mathematical claims or proofs, method or experiment design, implementation, data analysis, or interpretation of results. The authors reviewed every edit and take full responsibility for the manuscript.

REPRODUCIBILITY STATEMENT

The appendices document the training algorithm, experimental settings, benchmark protocols, hyperparameters, debate prompts, resource use, and theoretical proofs. They also specify the final-checkpoint evaluation protocol, decoding uncertainty, and compute-controlled comparisons.

REFERENCES

Rishabh Agarwal et al. On-policy distillation of language models: Learning from self-generated mistakes. In *International Conference on Learning Representations*, 2024.

Aayush Aluru et al. SMAGDi: Socratic multi agent interaction graph distillation for efficient high accuracy reasoning. *arXiv preprint arXiv:2511.05528*, 2025.

Jacob Austin et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik Narasimhan. τ^2-bench: Evaluating conversational agents in a dual-control environment. *arXiv preprint arXiv:2506.07982*, 2025.

Justin Chih-Yao Chen, Swarnadeep Saha, and Mohit Bansal. ReConcile: Round-table conference improves reasoning via consensus among diverse LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, 2024a.

Justin Chih-Yao Chen, Swarnadeep Saha, Elias Stengel-Eskin, and Mohit Bansal. MAGDi: Structured distillation of multi-agent interaction graphs improves reasoning in smaller language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 7220–7235, 2024b.

Qianglong Chen, Feng Ji, Feng-Lin Li, Guohai Xu, Ming Yan, Ji Zhang, and Yin Zhang. AMTSS: An adaptive multi-teacher single-student knowledge distillation framework for multilingual language inference. *arXiv preprint arXiv:2305.07928*, 2023.

Yiqun Chen et al. Improving retrieval-augmented generation through multi-agent reinforcement learning. In *Advances in Neural Information Processing Systems*, 2025.

DeepSeek-AI et al. DeepSeek-V4: Towards highly efficient million-token context intelligence. *arXiv preprint arXiv:2606.19348*, 2026.

Yilun Du et al. Improving factuality and reasoning in language models through multiagent debate. In *Proceedings of the 41st International Conference on Machine Learning*, 2024. URL `https://proceedings.mlr.press/v235/du24e.html`.

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.

Yuxian Gu et al. MiniLLM: Knowledge distillation of large language models. In *International Conference on Learning Representations*, 2024.

Etash Guha, Ryan Marten, Sedrick Keh, Negin Raoof, Georgios Smyrnis, Hritik Bansal, Marianna Nezhurina, Jean Mercat, Trung Vu, Zayne Sprague, et al. OpenThoughts: Data recipes for reasoning models. *arXiv preprint arXiv:2506.04178*, 2025.

Wei He, Yueqing Sun, Hongyan Hao, Xueyuan Hao, Zhikang Xia, Qi Gu, Chengcheng Han, Dengchang Zhao, Hui Su, Kefeng Zhang, et al. VitaBench: Benchmarking LLM agents with versatile interactive tasks in real-world applications. *arXiv preprint arXiv:2509.26490*, 2025.

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

Jonas Hübotter et al. Reinforcement learning via self-distillation. *arXiv preprint arXiv:2601.20802*, 2026.

Naman Jain et al. LiveCodeBench: Holistic and contamination-free evaluation of large language models for code. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/94074dd5a072d28ff75a76dabed43767-Abstract-Conference.html.

Ijun Jang, Jewon Yeom, Juan Yeo, Hyunggyu Lim, and Taesup Kim. Stable on-policy distillation through adaptive target reformulation. In *Findings of the Association for Computational Linguistics: ACL 2026*, 2026. URL https://aclanthology.org/2026.findings-acl.2094/.

Woogyeol Jin, Taywon Min, Yongjin Yang, Dennis Wei, Yi Zhou, Swanand Ravindra Kadhe, Nathalie Baracaldo, and Kimin Lee. Entropy-aware on-policy distillation of language models. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://research.ibm.com/publications/entropy-aware-on-policy-distillation-of-language-models--1.

Yoon Kim and Alexander M. Rush. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 1317–1327, 2016.

Weizhen Li, Jianbo Lin, Zhuosong Jiang, Jingyi Cao, Xinpeng Liu, Jiayu Zhang, Zhenqiang Huang, Qianben Chen, Weichen Sun, Qiexiang Wang, Hongxuan Lu, Tianrui Qin, Chenghao Zhu, Yi Yao, Shuying Fan, Xiaowan Li, Tiannan Wang, Pai Liu, King Zhu, He Zhu, Dingfeng Shi, Piaohong Wang, Yeyi Guan, Xiangru Tang, Minghao Liu, Yuchen Eleanor Jiang, Jian Yang, Jiaheng Liu, Ge Zhang, and Wangchunshu Zhou. Chain-of-agents: End-to-end agent foundation models via multi-agent distillation and agentic RL. *arXiv preprint arXiv:2508.13167*, 2025.

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

Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Shuming Shi, and Zhaopeng Tu. Encouraging divergent thinking in large language models through multi-agent debate. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 17889–17904. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.emnlp-main.992.

Jianhua Lin. Divergence measures based on the shannon entropy. *IEEE Transactions on Information Theory*, 37(1):145–151, 1991.

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 *Advances in Neural Information Processing Systems*, 2023.

Weiwen Liu et al. ToolACE: Winning the points of LLM function calling. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/663865ea167425c6c562cb0b6bcf76c7-Abstract-Conference.html.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

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

Shishir G. Patil, Huanzhi Mao, Fanjia Yan, Charlie Cheng-Jie Ji, Vishnu Suresh, Ion Stoica, and Joseph E. Gonzalez. The berkeley function calling leaderboard (BFCL): From tool use to agentic evaluation of large language models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 48371–48392, 2025.

Emiliano Penaloza, Dheeraj Vattikonda, Nicolas Gontier, Alexandre Lacoste, Laurent Charlin, and Massimo Caccia. Privileged information distillation for language models. *arXiv preprint arXiv:2602.04942*, 2026.

Qwen Team. Qwen3.5 model family. `https://huggingface.co/Qwen`, 2026. Open-weight release.

Zhanming Shen, Zeyu Qin, Zenan Huang, Hao Chen, Jiaqi Hu, Yihong Zhuang, Guoshan Lu, Gang Chen, and Junbo Zhao. Merge-of-thought distillation. *arXiv preprint arXiv:2509.08814*, 2025.

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

Zhiheng Xi et al. The rise and potential of large language model based agents: A survey. *Science China Information Sciences*, 68:121101, 2025. doi: 10.1007/s11432-024-4222-0.

An Yang et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Chenxu Yang et al. Self-distilled RLVR. *arXiv preprint arXiv:2604.03128*, 2026a.

Wenkai Yang et al. Learning beyond teacher: Generalized on-policy distillation with reward extrapolation. *arXiv preprint arXiv:2602.12125*, 2026b.

Tianzhu Ye et al. On-policy context distillation for language models. *arXiv preprint arXiv:2602.12275*, 2026.

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*, 2026.

Xiaofeng Zhou, Heyan Huang, and Lizi Liao. Debate, reflect, and distill: Multi-agent feedback with tree-structured preference optimization for efficient language model enhancement. In *Findings of the Association for Computational Linguistics: ACL 2025*, 2025. URL `https://aclanthology.org/2025.findings-acl.475/`.

A Algorithm Pseudocode

Algorithm 1 uses $M_{\max}=1$ for code generation and $M_{\max}=10$ for agentic rollouts. The realized trajectory length M can be shorter than this cap. The main experiments use $U=100$ optimizer updates. For agentic training, τ_0 is an initial task context obtained from ToolACE; subsequent observations are generated online by the tool simulator. Teachers first debate the current state s_m to produce $\mathcal{H}_m^R$, then evaluate the sampled student action a_m through privileged force-decoding. The debate concerns the decision at s_m, not the tokens of the student's draft.

Algorithm 1 MAD-OPD: Multi-Agent Debate-Driven On-Policy Distillation

Require: Teachers $\{T_1, \dots, T_K\}$, student π_θ, data $\mathcal{D}$, debate rounds R, divergence D, max steps $M_{\max}$, update budget U, confidence temperature τ_{conf}, learning rate η

repeat
  Sample batch $\mathcal{B}$ from $\mathcal{D}$
  for each $x \in \mathcal{B}$ **do**
    $\mathcal{L}_x \leftarrow 0$; $\tau \leftarrow \tau_0$ {initial task context; $\tau_0 = \emptyset$ for single-step tasks}
    for step $m = 1, \dots, M_{\max}$ **do**
      $a_m \sim \pi_\theta(\cdot \mid x, \tau)$ {on-policy student action}
      for round $r = 1, \dots, R$ **do**
        for teacher $k = 1, \dots, K$ **in parallel do**
          $h_r^k \sim p_{T_k}(\cdot \mid x, \tau, \{h_{r'}^j\}_{j, r' < r})$
        end for
      end for
      $\mathcal{H}_m^R \leftarrow \{h_r^k\}_{k,r}$ {privileged context}
      $\forall k: \; c_k \leftarrow \text{parse_confidence}(h_R^k)$; $w_k \leftarrow \text{softmax}_k\big(c_\cdot / (100\,\tau_{\text{conf}})\big)$ {confidence weights}
      $\mathbf{p}_{T_k}^{(t)} \leftarrow p_{T_k}(\cdot \mid x, \tau, \mathcal{H}_m^R, a_{m,<t}) \quad \forall k, t$ {force-decode **with** $\mathcal{H}_m^R$}
      $\mathbf{p}_S^{(t)} \leftarrow \pi_\theta(\cdot \mid x, \tau, a_{m,<t}) \quad \forall t$ {student **without** $\mathcal{H}_m^R$}
      $\mathcal{L}_x \leftarrow \mathcal{L}_x + \frac{1}{|a_m|} \sum_t \sum_k w_k \cdot D\big(\mathbf{p}_{T_k}^{(t)} \,\|\, \mathbf{p}_S^{(t)}\big)$
      if a_m is a final answer **then**
        break
      end if
      $o_m \sim P^{\text{env}}(\cdot \mid x, \tau, a_m)$; $\tau \leftarrow \tau \oplus (a_m, o_m)$
      if the environment terminates or the context budget is reached **then**
        break
      end if
    end for
  end for
  $\theta \leftarrow \theta - \eta \nabla_\theta \frac{1}{|\mathcal{B}|} \sum_x \mathcal{L}_x$ {update π_θ only}
until U optimizer updates have been completed

B Experimental Details

This appendix gives the training, infrastructure, data, and evaluation details for all experiments.

B.1 Training Configuration

The main experiments use non-thinking mode (`enable_thinking=False`) during training and evaluation. Optimization uses AdamW (Loshchilov & Hutter, 2019) with β_1=0.9, β_2=0.999, weight decay 0.01, learning rate 1×10^{-5} with cosine decay and 5% warmup. Effective batch size is 128. Agentic tasks cap student input at 16,384 tokens and student generation at 4,096 new tokens per step; code tasks cap student input at 4,096 tokens and student generation at 16,384 new tokens per problem. The main experiments train each method for 100 optimizer steps and evaluate its final checkpoint. The 30B-A3B teacher is the `Qwen3-30B-A3B-Instruct-2507` release; all other Qwen3/Qwen3.5 teachers and students use the standard Instruct checkpoints of the corresponding size. All Qwen3 and Qwen3.5 checkpoints are released under Apache 2.0 license.

B.2 Infrastructure

All experiments run on 8× NVIDIA H20 GPUs with DeepSpeed ZeRO-3. Two single-GPU teacher processes serve each run. vLLM generates debate responses, and a sidecar extracts the full token-level logit distribution for distillation. We set `max_model_len`= 32,768 for vLLM and use a maximum sequence length of 40,960 for the sidecar to accommodate the input, debate transcript, and student action.

A 4B-student agentic run takes approximately 16 hours. Code training takes approximately 32 hours because of its longer generation length. Across the six teacher–student configurations, baselines, ablations, and 16-seed code evaluations, the project used approximately 8,000 H20 GPU-hours.

B.3 Training Data

Agentic data. Training uses the ToolACE training set (Liu et al., 2025), released under Apache 2.0, with approximately 16K tool-use task contexts and tool schemas. Recorded dialogue prefixes initialize the rollout state τ_0. At each decision step, the student samples a tool action from the current state, and the teachers debate that state and force-decode the sampled action. Our ToolACE-based online simulator processes the action and returns a tool response o_m, which is appended to the context for the next decision. This loop continues until termination, applying step-wise OPAD supervision throughout the interaction.

Code data. We sample 30K problems from OpenThoughts3 (Guha et al., 2025), released under Apache 2.0. For each problem, the student generates an on-policy code completion. Both teachers then produce token-level distributions conditioned on the student prefix and debate history, and their confidence-weighted divergences provide the training signal. Agentic and code tasks use separate checkpoints and training sets.

B.4 Online Rollout Protocol

ToolACE provides the task contexts and tool schemas. Its data-generation framework separates user, assistant, and tool roles (Liu et al., 2025); our online rollout uses the student as the assistant and Claude-opus-4-6 to simulate both tool responses and user replies.

Horizon and stopping rules. We cap each agentic rollout at $M_{\max}{=}10$ student decision turns. A turn is one student generation, including a tool action, clarification request, or final answer. Parallel tool calls within one generation count as one decision turn. A rollout ends on a final answer, an environment terminal state, or the turn or context limit. EOS ends the current generation; a tool call followed by EOS still receives an environment response. The realized length $M \leq 10$ counts generated student turns, excluding the initial dialogue prefix. The outer training loop runs for 100 optimizer updates and uses the final checkpoint; equal-time comparisons follow their stated budgets.

Tool responses and invalid calls. For action-dependent simulation, the tool-response role conditions on the sampled call, tool schema, and accumulated environment state. The protocol validates call syntax, function names, required arguments, and argument types before generating tool responses. Invalid calls return a structured error without changing the environment state. They count toward the ten-turn budget and retain token-level supervision, allowing the student to recover on a later turn.

Clarification responses. Clarification requests are routed to the user role, which answers from the task specification and dialogue history. This role supplies requested task information rather than a reference action or solution. Tool and user responses remain outside the student loss, and neither simulator role receives the privileged teacher debate.

B.5 Baseline Setup

Base. The undistilled instruct checkpoint of the corresponding student size, evaluated directly. **OPD** (Agarwal et al., 2024). Single-teacher on-policy distillation using the stronger teacher of each pair (e.g., Qwen3-14B for the 14B+8B pair); divergence and decoding match MAD-OPD. **MT-OPD**. Naive multi-teacher OPD: both teachers produce token-level distributions independently and the loss aggregates them with uniform weights ($w_k = 1/K$); no inter-teacher debate. **MT-SeqKD** (Kim & Rush, 2016). Off-policy multi-teacher sequence distillation routed by a 7:3 strong-to-weak teacher mix: the strong teacher generates 70% of the training sequences and the weak teacher 30%, mixed at the sequence level. The student is trained with sequence-level cross-entropy on the routed teacher outputs.

B.6 Debate Configuration

Each debate uses $K{=}2$ teachers and $R{=}2$ rounds. In round 1, the teachers independently answer from the task state without seeing the student action. In round 2, each teacher sees both round-1 responses with speaker identities and revises its answer. We use temperature 0.7 and cap each response at $4{,}096$ new tokens, giving a maximum transcript length of $16{,}384$ tokens. After round 2, each teacher reports

Table 2: Complete training and evaluation configuration.

Parameter	Agentic Tasks	Code Tasks
Training		
Optimizer	AdamW (β_1=0.9, β_2=0.999)	AdamW (β_1=0.9, β_2=0.999)
Weight decay	0.01	0.01
Learning rate	1×10^{-5}, cosine, 5% warmup	1×10^{-5}, cosine, 5% warmup
Effective batch size	128	128
Max input length	16384	4096
Max new tokens	4096	16384
Data	ToolACE ($\sim$16K)	OpenThoughts3 (30K)
Thinking mode	Non-thinking	Non-thinking
Student sampling top-k	20	20
Student repetition penalty	1.2	1.2
Distillation		
Teachers (K)	2	2
Debate rounds (R)	2	2
Debate temperature	0.7	0.7
Divergence (D)	$\mathrm{JSD}_{0.5}$	$D_{\mathrm{KL}}^{\leftarrow}$
Logit vocabulary	Full vocabulary	Full vocabulary
Logit precision	BF16	BF16
Infrastructure		
Training GPUs	8$\times$ NVIDIA H20 (ZeRO-3)	8$\times$ NVIDIA H20 (ZeRO-3)
Teacher debate serving	vLLM (single H20)	vLLM (single H20)
Teacher vLLM max-model-len	32768	32768
Logit extraction	Sidecar (single H20)	Sidecar (single H20)
Evaluation		
Temperature	0.7	0.7
Top-p	0.8	0.8
Max generation length	4096	16384
Decoding	Sampling	Sampling (pass@1)
BoN	–	N=16

an integer confidence score from 0 to 100. We divide the score by 100 and normalize the two values with a softmax at τ_{conf}=1.0 as in Eq. 7c. The prompt is identical across configurations and model families. App. C.9 compares $R \in \{0, 1, 2, 3\}$.

B.7 Evaluation Protocol

We evaluate the final checkpoints after 100 optimizer steps, using the same training length across methods. Each agent checkpoint is evaluated on BFCL-v4, τ^2-Bench, and VitaBench; each code checkpoint is evaluated on LCB-v6 and MBPP+. The equal-time comparisons in App. C.1 and App. C.2 use their stated training budgets.

Code pass@1 scores are averaged over 16 random seeds, with standard deviations reported in Tables 1 and 8. We report the official BFCL-v4 overall score, the three-domain mean for τ^2-Bench, and task success on the full VitaBench benchmark.

Evaluation uses temperature 0.7 and top-$p = 0.8$, following the corresponding values in the Qwen3 non-thinking configuration (Yang et al., 2025). The maximum generation length is $4{,}096$ tokens for agentic tasks and $16{,}384$ for code. Evaluation uses `max_model_len`$=32{,}768$, increased to $40{,}960$ for BFCL-v4. App. C.7 also reports BoN@16 from 16 samples per problem. For τ^2-Bench and VitaBench, Claude-opus-4-6 serves as the user simulator, with each episode capped at 300 environment steps. Sections B.9–B.13 give the benchmark-specific protocols.

B.8 Hyperparameter Summary

Table 2 lists the complete configuration. Agentic training uses $\mathrm{JSD}_{0.5}$ and a maximum of 4,096 new tokens; code training uses reverse KL and a maximum of 16,384 new tokens. Student sampling during training uses temperature 0.7, top-p 0.8, top-k 20, and repetition penalty 1.2.

B.9 Benchmark: BFCL-v4

Berkeley Function-Calling Leaderboard v4 (BFCL-v4) (Patil et al., 2025) evaluates function calling and agentic tool use. We use the full official evaluation and report its overall score, which combines Agentic, Multi-Turn, Live, Non-Live, and Hallucination scores with weights of 40%, 30%, 10%,

10%, and 10%, respectively. Evaluation follows the official procedure for each category, including AST matching for single-turn function calls and state- and response-based checks for multi-turn tasks. The benchmark is available at `https://github.com/ShishirPatil/gorilla/tree/main/berkeley-function-call-leaderboard` under Apache 2.0 license.

B.10 BENCHMARK: τ^2-BENCH

τ^2-Bench (Barres et al., 2025) extends the airline and retail domains of the original τ-bench with a dual-control **telecom** domain and a compositional task generator. It evaluates multi-turn customer-service agents in three domains:

- **Retail**: e-commerce account management (order returns, exchanges, refunds);
- **Airline**: flight reservation modifications, seat selection, fare changes;
- **Telecom**: account management, service plan changes, troubleshooting flows.

Each task provides a domain-specific toolset and policy document. The agent must complete the request through tool calls while following the policy and tracking changes across the conversation. We use **Claude-opus-4-6** as the user simulator and cap each episode at 300 environment steps. The official scorer checks whether the final database state exactly matches the target state. The score in Table 1 is the pass@1 success rate averaged over the three domains. We use the repository release with 75+ airline and retail task fixes and the original τ^2 telecom implementation; the voice and banking-knowledge modalities are excluded. The benchmark is available at `https://github.com/sierra-research/tau2-bench` under the MIT license.

B.11 BENCHMARK: VITABENCH

VitaBench (He et al., 2025) evaluates interactive agents on life-service tasks with 66 tools across three domains:

- **Food delivery**: restaurant search, menu navigation, order placement, delivery tracking;
- **In-store consumption**: in-shop ordering, payment, inventory checks, loyalty programs;
- **Online travel agency (OTA)**: flight/hotel/itinerary search and booking, modification, refunds.

We evaluate all 400 VitaBench tasks: 100 cross-scenario tasks and 100 single-scenario tasks each from delivery, in-store consumption, and OTA. Each task is evaluated once. The tasks require tool use, clarification of ambiguous instructions, and multi-turn intent tracking. Evaluation uses the official rubric-based sliding-window scorer. We use **Claude-opus-4-6** as the user simulator and cap each episode at 300 environment steps. The Vita score in Table 1 is the task success rate across all 400 tasks. The benchmark is available at `https://vitabench.github.io/` under the MIT license.

B.12 BENCHMARK: LIVECODEBENCH V6

LiveCodeBench v6 (LCB-v6) (Jain et al., 2025) is a timestamped code-generation benchmark built from LeetCode, AtCoder, and Codeforces problems. Release windows filter problems by publication date to reduce overlap with model training data. We use version v6. Each problem has a natural-language description and hidden tests, which are executed with a per-test timeout. The primary metric is pass@1 with temperature 0.7 and top-p 0.8. We also report BoN@16 using 16 samples per problem at temperature 0.7. Scores are averaged over 16 random seeds, with standard deviations shown in subscripts. The benchmark is available at `https://github.com/LiveCodeBench/LiveCodeBench` under Apache 2.0.

B.13 BENCHMARK: MBPP+

MBPP+ (Liu et al., 2023) is the EvalPlus extension of **MBPP** (Austin et al., 2021). EvalPlus expands the original test suites through differential testing and mutation-based input synthesis. The additional tests expose failures such as off-by-one errors and mishandled edge cases. MBPP+ contains 378 sanitized Python problems with a description, function signature, and test suite. We

Table 3: Controlled non-thinking results on Qwen3-14B+8B→4B. Code values are mean ± standard deviation over 16 random seeds.

Method	BFCL-v4	τ^2	Vita	Ag-Avg	LCB-v6	MBPP+	Co-Avg	Overall
OPD-Long	25.29	29.53	16.75	23.86	$26.27_{\pm 0.51}$	$63.96_{\pm 0.85}$	45.12	32.36
MT-NoComm-2R	25.53	29.23	17.50	24.09	$24.01_{\pm 0.62}$	$61.54_{\pm 1.02}$	42.78	31.56
ST-SelfRefine-2x2	25.82	29.04	17.50	24.12	$25.52_{\pm 0.92}$	$62.82_{\pm 0.83}$	44.17	32.14
OPD-Teacher-CoT	25.64	29.83	17.00	24.16	$25.80_{\pm 0.74}$	$63.22_{\pm 0.79}$	44.51	32.30
MAD-OPD	**27.17**	**31.42**	**18.75**	**25.78**	$\mathbf{29.72}_{\pm 0.73}$	$\mathbf{66.30}_{\pm 0.62}$	**48.01**	**34.67**

Table 4: Equal-time non-thinking agent comparison. Both runs use the same recorded hardware allocation and 20-hour elapsed-time budget.

Method	Time	Updates	Ag-Avg
OPD-Long	20 h	205	23.86
MAD-OPD	20 h	117	**25.78**

report pass@1 and BoN@16 under the same 16-seed protocol as LCB-v6. The benchmark is available at `https://github.com/evalplus/evalplus` under Apache 2.0.

C Additional Experiment Results

This appendix consolidates supplementary empirical evidence; each subsection indicates which research question (Sec. 5) it supports.

C.1 Controlled Comparisons of Communication and Compute

We study the Qwen3-14B+8B→4B configuration with four controls, reported in Table 3. **MT-NoComm-2R** preserves the teachers, two-round call structure, force-decodes, objective, optimizer, and confidence aggregation while preventing peer-response visibility. **ST-SelfRefine-2x2** assigns four generations to the stronger 14B teacher through two proposal–revision branches. **OPD-Long** allocates the same recorded agent hardware and elapsed-time budget to single-teacher OPD. **OPD-Teacher-CoT** supplies a longer teacher-only reasoning trace before force-decoding the same non-thinking student action.

In the controlled comparison, MAD-OPD reaches 25.78 Ag-Avg, 48.01 Co-Avg, and 34.67 Overall. Enabling peer-response visibility improves Overall by 3.11 points over MT-NoComm-2R, which retains the same teachers and two-round call structure. MAD-OPD also exceeds ST-SelfRefine-2x2 by 2.53 Overall points and OPD-Teacher-CoT by 1.62 Ag-Avg and 3.50 Co-Avg. OPD-Teacher-CoT uses $1.34\times$ and $1.65\times$ as many teacher-only privileged tokens in the agent and code domains, respectively. These comparisons attribute the gain to inter-teacher communication beyond additional teacher calls or longer privileged reasoning.

Within the common agent-training budget, MAD-OPD improves Ag-Avg by 1.92 points despite completing 88 fewer updates. Code checkpoints are trained separately.

C.2 Native-Thinking Generalization

We repeat the representative configuration with native thinking enabled for the student and every teacher during training and evaluation. OPD and MAD-OPD share initialization, ToolACE data, JSD objective, loss mask, parser, optimizer, and recorded hardware allocation. Only the final tool action is exposed to the environment; private reasoning remains internal.

Table 5 shows that MAD-OPD exceeds OPD on all five benchmarks, improving Ag-Avg by 1.76, Co-Avg by 2.24, and Overall by 1.95 points. The matched agent runs each use a 40-hour budget; MAD-OPD completes 100 updates and OPD completes 137. The code checkpoints are trained separately.

Table 5: Native-thinking results. Code values are mean ± standard deviation over 16 random seeds.

Method	BFCL-v4	τ^2	Vita	Ag-Avg	LCB-v6	MBPP+	Co-Avg	Overall
Base + thinking	53.39	27.41	18.75	33.18	$34.72_{\pm 1.03}$	$69.84_{\pm 0.92}$	52.28	40.82
OPD-Student-CoT	58.74	37.02	23.00	39.59	$40.21_{\pm 1.03}$	$70.38_{\pm 0.92}$	55.30	45.87
MAD-OPD-Student-CoT	**62.86**	**37.42**	**23.75**	**41.34**	$\mathbf{41.84}_{\pm 0.84}$	$\mathbf{73.22}_{\pm 0.78}$	**57.53**	**47.82**

Table 6: **Component analysis** on the 14B+8B→4B setting. **Multi-T** denotes multi-teacher aggregation, **On-Policy** student-rollout supervision, **Debate** inter-teacher debate, and **Conf. Wt.** confidence weighting. **Bold**: best; underline: second best.

Method	Multi-T	On-Policy	Debate	Conf. Wt.	Ag-Avg	Co-Avg	Overall
Base					18.61	41.50	27.77
MT-SeqKD	✓				21.44	40.77	29.17
OPD		✓			23.24	44.41	31.71
MT-OPD	✓	✓			24.15	41.97	31.28
MAD-OPD w/o Conf. Wt.	✓	✓	✓		25.85	46.60	34.15
MAD-OPD	✓	✓	✓	✓	**27.63**	**48.65**	**36.03**

Table 7: **Divergence ablation** on MAD-OPD with 14B+8B→4B. JSD leads on agentic tasks, and reverse KL leads on code. Code pass@1 is averaged over 16 seeds; subscripts report standard deviation.

	Agentic Tasks				**Code Tasks**		
Divergence D	BFCL-v4	τ^2	Vita	Ag-Avg	LCB-v6	MBPP+	Co-Avg
Fwd KL ($D_{\mathrm{KL}}^{\rightarrow}$)	23.70	26.24	15.25	21.73	$24.29_{\pm 0.73}$	$57.03_{\pm 1.13}$	40.66
Rev KL ($D_{\mathrm{KL}}^{\leftarrow}$)	25.02	25.85	15.75	22.21	$\mathbf{29.87}_{\pm 0.56}$	$\mathbf{67.42}_{\pm 0.92}$	**48.65**
JSD (JSD_{β})	**28.39**	**33.74**	**20.75**	**27.63**	$24.92_{\pm 0.92}$	$64.50_{\pm 0.95}$	44.71

C.3 Teacher Computation

The quantity KR counts teacher-generation calls, while the full teacher cost also depends on prompt lengths, completion lengths, and K privileged force-decodes. With $K{=}2$ and $R{=}2$, each response is capped at 4,096 tokens and the full transcript at 16,384 tokens. The transcript is visible only to the teachers. Parameter–token proxies omit attention costs, KV-cache behavior, batching, communication, and serving utilization, so we report measured GPU-hours in App. B.2.

C.4 Component Analysis: Full Numbers (RQ3)

Table 6 reports the per-component ablation underlying Figure 4(a) on the 14B+8B→4B configuration. The rows compare baselines and component variants.

Confidence weighting. In the 14B+8B→4B experiment, confidence weighting improves Ag-Avg from 25.85 to 27.63 ($+1.78$ points) and Co-Avg from 46.60 to 48.65 ($+2.05$ points) over uniform post-debate weighting. Equal confidence scores recover uniform post-debate weighting.

C.5 Divergence Ablation: Full Numbers (RQ4)

Table 7 compares the three divergences on the 14B+8B→4B configuration with all other hyperparameters fixed. JSD leads on agentic tasks, while reverse KL leads on code.

C.6 Student Outperforms Teachers on LCB-v6 (RQ1)

Figure 5 compares accuracy and output length for the 14B+8B→4B configuration on LCB-v6. Its longer solutions expose clear token-cost differences between methods.

The 4B student trained with MAD-OPD reaches 29.87% pass@1, exceeding the 14B teacher (25.57%) and the 8B teacher (20.93%): a 4.30-point gain over the strongest teacher despite using $3.5\times$ fewer parameters and at competitive token cost vs. all distillation baselines. The advantage extends beyond pass@1: under BoN@16 the same 4B student reaches 43.43%, outperforming the

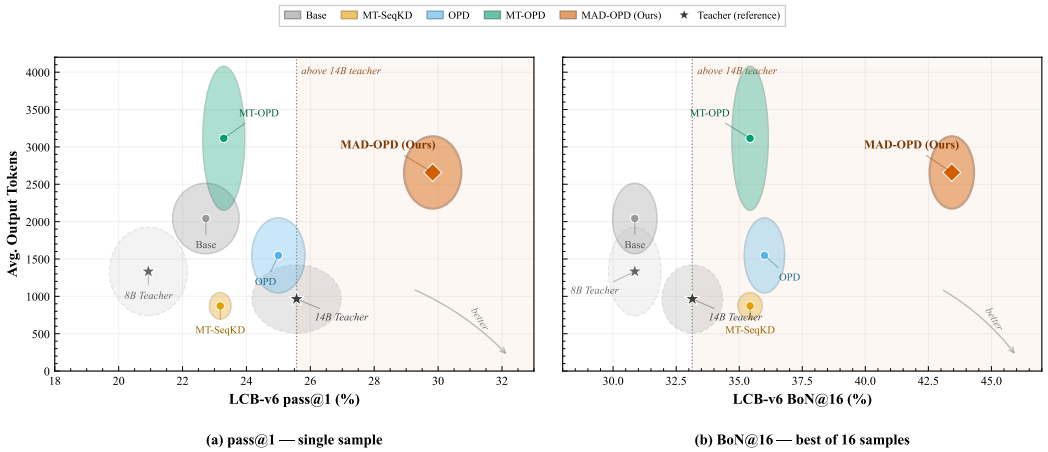


Figure 5: **Accuracy and output length on LCB-v6** for 14B+8B→4B over 16 random seeds. Ellipses span one standard deviation, markers show seed means, and gray stars mark the teachers. **(a)** MAD-OPD reaches 29.87% pass@1, 4.30 points above the 14B teacher and 4.87 above OPD. **(b)** Its BoN@16 reaches 43.43%, exceeding the 14B teacher by 10.29 points and OPD by 7.43.

Table 8: **Code BoN@16 results** from 16 samples per problem. **Bold**: best per configuration; underline: second best. Teacher results are in Table 9.

	Qwen3 14B+8B		Qwen3 32B+30B-A3B			Qwen3.5 27B+9B
Method	→1.7B	→4B	→4B	→8B	→14B	→4B
LCB-v6 BoN@16						
MT-SeqKD	28.00	35.43	36.57	**40.57**	42.86	**49.71**
OPD	**31.43**	36.00	36.00	36.00	38.92	41.14
MT-OPD	26.29	35.43	38.86	39.43	47.43	49.41
MAD-OPD	**31.43**	**43.43**	**43.43**	39.43	**49.71**	**49.71**
MBPP+ BoN@16						
MT-SeqKD	61.90	70.37	71.62	69.58	77.78	66.67
OPD	66.67	73.81	67.73	73.54	80.95	75.13
MT-OPD	65.18	67.72	69.58	75.40	80.42	81.48
MAD-OPD	**67.73**	**74.34**	**73.81**	**76.46**	**81.69**	**81.69**

14B teacher's BoN@16 (33.14%) by 10.29 points. No other distillation baseline crosses the 14B-teacher pass@1 reference: OPD lands at 25.00%, MT-OPD at 23.29%, MT-SeqKD at 23.18%. The result shows that confidence-weighted multi-teacher debate produces supervision that exceeds the strongest single teacher in its own pool, both in single-shot and multi-sample selection regimes.

C.7 Best-of-N Results for Code Benchmarks (RQ1)

For BoN@16, we sample 16 solutions per problem at temperature 0.7 and select the candidate that passes the most evaluation tests from the corresponding benchmark (LCB-v6 or MBPP+). This reports oracle selection using the benchmark tests. Table 8 reports results across all six configurations. On LCB-v6, MAD-OPD is best or tied best in five configurations; MT-SeqKD leads the 32B+30B→8B setting by 1.14 points. On MBPP+, MAD-OPD ranks first in every configuration.

C.8 Training Loss Stability under Different Divergences (RQ4)

Figure 6 shows per-step training losses under the three divergences for 14B+8B→4B. On agentic tasks, reverse-KL loss spans two orders of magnitude, while JSD remains within a narrower range. On code, reverse-KL loss decreases smoothly. Table 7 reports the corresponding task scores.

C.9 Effect of Debate Rounds R (RQ3)

Figure 7 compares $R \in \{0, 1, 2, 3\}$ on the 14B+8B→4B configuration. At R=0, training uses uniform MT-OPD without generated teacher context. At R=1, each teacher generates an independent

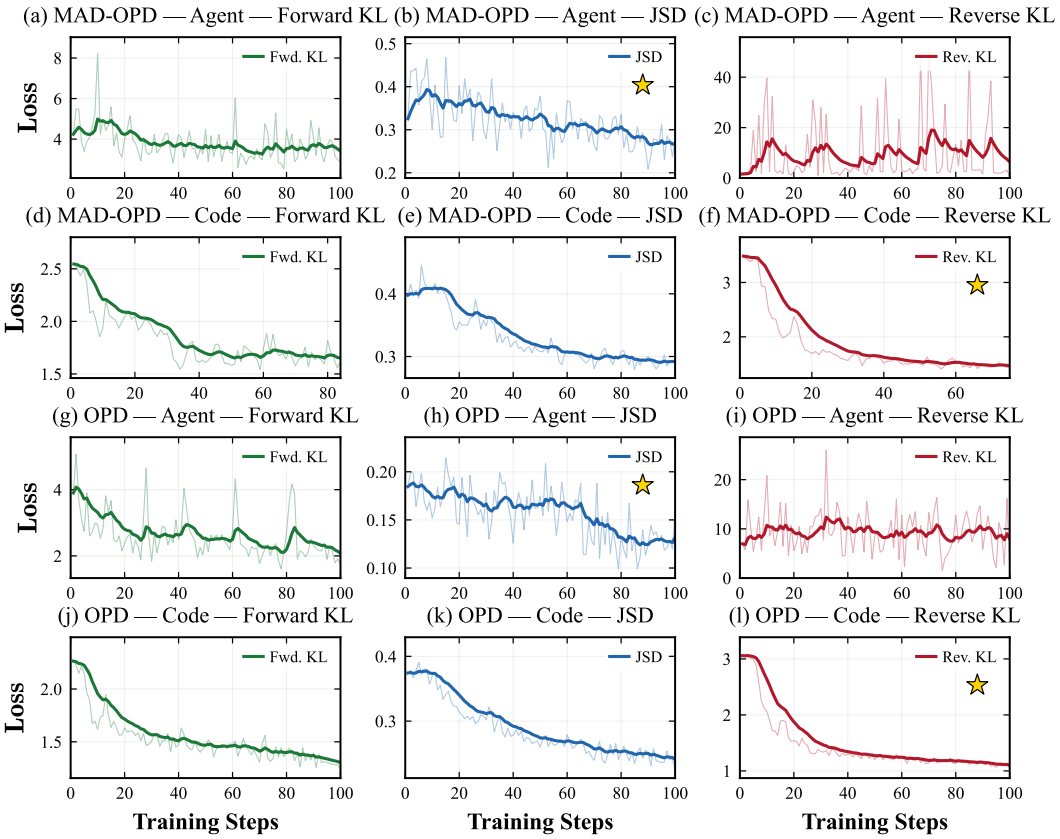


Figure 6: **Training loss curves** for MAD-OPD and OPD under forward KL, JSD, and reverse KL in the 14B+8B→4B setting. Rows 1–2 show MAD-OPD, and rows 3–4 show OPD; agentic and code tasks alternate by row. Curves cover the first 100 gradient steps. Light lines show per-step loss, and solid lines show an exponential moving average with $\alpha = 0.15$. Reverse KL is highly variable on agentic tasks, JSD is smoother, and reverse KL converges smoothly on code.

response and confidence score; its response provides privileged context for its own force-decoding, and the confidence scores determine the teacher weights. Teachers exchange responses for revision from R=2 onward. The default R=2 achieves the highest score on every benchmark. At R=3, the scores are BFCL-v4 24.64, τ^2-Bench 28.81, VitaBench 16.75, LCB-v6 26.93, and MBPP+ 63.42, corresponding to Ag-Avg 23.40, Co-Avg 45.18, and Overall 32.11. Compared with R=2, a third round reduces Ag-Avg by 4.23 points and Co-Avg by 3.47 points.

C.10 Teacher Base Performance Reference

Table 9 consolidates the undistilled performance of every teacher used in the six configurations of Sec. 5.1. These scores provide the single-teacher reference levels that MAD-OPD aims to exceed through debated supervision. The 30B-A3B teacher refers to the `Qwen3-30B-A3B-Instruct-2507` release; all other teachers use the standard Instruct checkpoints of the corresponding size (App. B).

D Detailed Proofs

This appendix derives the dense OPD gradient, the JSD bounds, and the geometric consensus induced by the multi-teacher reverse-KL objective. It also analyzes forward KL under multi-modal agentic supervision.

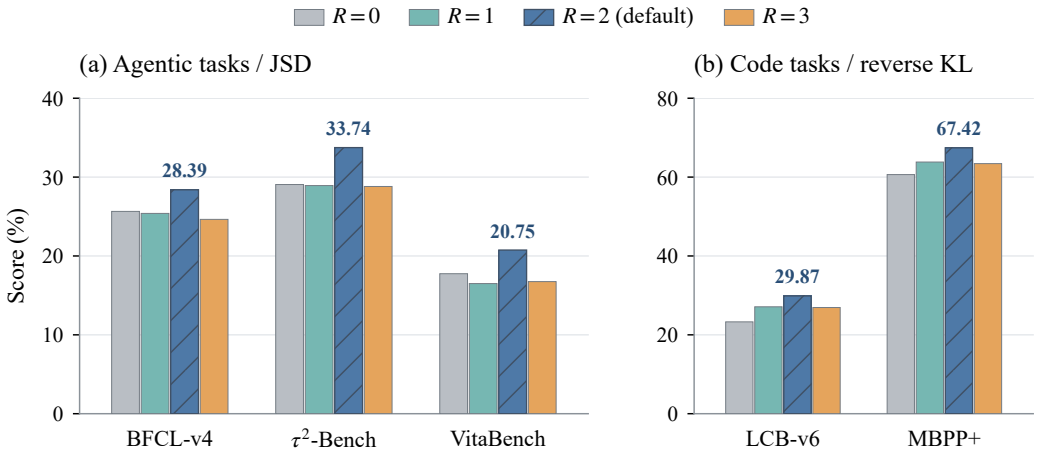


Figure 7: **Effect of debate rounds** on 14B+8B→4B. R=0 is uniform MT-OPD with no generated teacher context; R=1 adds independent teacher responses as privileged context and confidence weighting; R=2 adds one round of inter-teacher revision. Hatched blue bars mark the default R=2, which leads on all five benchmarks. Agentic and code tasks are shown in separate panels with zero-based y-axes.

Table 9: **Teacher base performance** across the five benchmarks. τ^2 reports the three-domain mean; Vita reports success across all 400 tasks. Code columns report pass@1 over 16 seeds and BoN@16 from 16 samples per problem.

	Agentic			Code pass@1		Code BoN@16	
Teacher	BFCL-v4	τ^2	Vita	LCB-v6	MBPP+	LCB-v6	MBPP+
Qwen3 family							
Qwen3-8B	28.94	30.08	17.25	$20.93_{\pm 0.76}$	$63.56_{\pm 0.61}$	30.86	70.90
Qwen3-14B	32.46	33.70	21.75	$25.57_{\pm 0.87}$	$75.13_{\pm 0.78}$	33.14	80.16
Qwen3-30B-A3B†	35.72	47.68	25.00	$36.00_{\pm 0.92}$	$77.87_{\pm 0.81}$	51.43	80.95
Qwen3-32B	38.01	48.39	25.75	$40.75_{\pm 0.84}$	$78.21_{\pm 0.65}$	56.37	81.75
Qwen3.5 family							
Qwen3.5-9B	36.97	49.33	11.75	$39.43_{\pm 1.13}$	$63.36_{\pm 1.06}$	55.43	79.37
Qwen3.5-27B	49.04	56.54	12.00	$43.43_{\pm 0.48}$	$72.21_{\pm 0.97}$	65.71	81.22

† `Qwen3-30B-A3B-Instruct-2507` release.

D.1 OPD Gradient Derivation

Proof. At each position t, $D(p\|q)$ depends on θ only through $q = \pi_\theta(\cdot|x, \hat{y}_{<t})$; the privileged teacher $p = \pi^*(\cdot|x, c, \hat{y}_{<t})$ is a stop-gradient target. The implementation also detaches the sampled rollout $\hat{y}$ and omits the score-function derivative through its sampling distribution. Conditioned on the sampled trajectory, the resulting dense gradient estimator is

$$g_{\text{dense}}(\theta) \triangleq \mathbb{E}_{x,\,\hat{y}\sim\pi_\theta}\left[\frac{1}{|\hat{y}|}\sum_{t=1}^{|\hat{y}|}\nabla_\theta D(p\,\|\,q)\right]. \tag{11}$$

This is the gradient of the stop-gradient rollout surrogate used by OPD, rather than the full derivative of Eq. 1 through $\hat{y} \sim \pi_\theta$. As an illustration, decomposing forward KL as $D_{\mathrm{KL}}^{\rightarrow}(p\|q) = H(p,q) - H(p)$ (with $H(p)$ constant in z) and applying the softmax Jacobian $\partial q(v)/\partial z_i = q(v)(\delta_{iv} - q(i))$ collapses the chain rule to

$$\nabla_{z_i} D_{\mathrm{KL}}^{\rightarrow}(p\|q) \;=\; -\sum_v p(v)(\delta_{iv} - q(i)) \;\overset{\sum p=1}{=}\; q(i) - p(i), \tag{12}$$

the standard "$q - p$" cross-entropy gradient with $|q(i) - p(i)| \leq 1$ uniformly. Stability thus reduces to the per-position bound $\|\nabla_{z_t} D\|_\infty$, derived per divergence in §D.2–D.3. □

D.2 Proof of Lemma 1 (JSD and Logit-Gradient Bounds)

Restatement. For any distributions p, q over $\mathcal{V}$ and any $\beta \in (0, 1)$, (i) $\mathrm{JSD}_\beta(p\|q) \in [0, H(\beta)]$ with $H(\beta) = -\beta \log \beta - (1-\beta)\log(1-\beta)$; (ii) $\|\nabla_z \mathrm{JSD}_\beta(p\|q)\|_\infty \leq 2$ uniformly over all p, q.

Proof. Let $m = \beta p + (1-\beta)q$ and $q(v) = \mathrm{softmax}(z)_v$, with $0 \log 0 := 0$; softmax gives $q(v) > 0$, hence $m(v) \geq (1-\beta)q(v) > 0$ everywhere.

Part 1: Loss boundedness. From $m(v) \geq \beta p(v)$ we have $\frac{p(v)}{m(v)} \leq \frac{1}{\beta}$, hence $\log \frac{p(v)}{m(v)} \leq \log \frac{1}{\beta}$, so

$$D_{\mathrm{KL}}(p\|m) \;=\; \sum_v p(v) \log \tfrac{p(v)}{m(v)} \;\leq\; \sum_v p(v) \log \tfrac{1}{\beta} \;=\; \log \tfrac{1}{\beta}, \tag{13}$$

and symmetrically $D_{\mathrm{KL}}(q\|m) \leq \log \frac{1}{1-\beta}$ from $m(v) \geq (1-\beta)q(v)$. Combining and writing $H(\beta) := -\beta \log \beta - (1-\beta)\log(1-\beta)$,

$$\mathrm{JSD}_\beta(p\|q) \;=\; \beta\, D_{\mathrm{KL}}(p\|m) + (1-\beta)\, D_{\mathrm{KL}}(q\|m) \;\leq\; H(\beta), \tag{14}$$

so $\mathrm{JSD}_{0.5} \leq \log 2$; the lower bound $\mathrm{JSD}_\beta \geq 0$ follows from $D_{\mathrm{KL}} \geq 0$.

Part 2: Gradient boundedness. Write $c = 1 - \beta$. Since

$$\mathrm{JSD}_\beta(p\|q) = H(m) - \beta H(p) - cH(q), \tag{15}$$

differentiation with respect to $q(v)$ gives

$$\frac{\partial \mathrm{JSD}_\beta(p\|q)}{\partial q(v)} = c \log \frac{q(v)}{m(v)}. \tag{16}$$

Combining this identity with the softmax Jacobian yields

$$\frac{\partial \mathrm{JSD}_\beta(p\|q)}{\partial z_i} = c\, q(i) \left[\log \frac{q(i)}{m(i)} - D_{\mathrm{KL}}(q\|m) \right], \tag{17}$$

which is Eq. 5. The positive log-ratio is bounded by $\log(1/c)$ because $m(i) \geq cq(i)$. For the negative case, $q(i)\log(m(i)/q(i)) \leq m(i) - q(i) \leq \beta$. Together with $D_{\mathrm{KL}}(q\|m) \leq \log(1/c)$,

$$\left| \frac{\partial \mathrm{JSD}_\beta}{\partial z_i} \right| \leq \max\{ c \log \tfrac{1}{c}, c\beta \} + c \log \tfrac{1}{c} \leq \frac{2}{e} < 1 < 2. \tag{18}$$

Thus the stated constant 2 holds uniformly over $\beta \in (0, 1)$ and all p, q, including mismatched supports. □

D.3 Proof of Proposition 1 (Per-Token Gradient Stability)

Restatement. Under the dense OPD gradient estimator (App. D.1), the per-token logit gradient satisfies: (i) JSD: $\|\nabla_{z_t} \mathrm{JSD}_\beta(p\|q)\|_\infty \leq 2$ uniformly, hence $\|\nabla_z \mathcal{L}_{\mathrm{JSD}}(\tau)\|_\infty \leq 2$ irrespective of trajectory length; (ii) Reverse KL: $\|\nabla_{z_t} D^{\leftarrow}_{\mathrm{KL}}(p\|q)\|_\infty$ has no uniform bound over positive teacher distributions; it diverges for fixed q with $q(i) \in (0, 1)$ as $p(i) \to 0$ while all other teacher probabilities stay bounded away from zero; (iii) Forward KL: $\|\nabla_{z_t} D^{\rightarrow}_{\mathrm{KL}}(p\|q)\|_\infty \leq 1$, with the empirical comparison discussed in App. D.5.

Proof. Throughout this proof, $z = (z_1, \ldots, z_N)$ denotes the *stacked* per-position pre-softmax logit vector, with $z_t \in \mathbb{R}^{|\mathcal{V}|}$. Bounds below are at the logit level; the parameter-level gradient $\nabla_\theta \mathcal{L}$ is the standard sum of per-position contributions through the shared θ.

For a trajectory τ of M steps, write either reduction as

$$\mathcal{L}_D(\tau) = \sum_{m=1}^{M} \sum_{t=1}^{|a_m|} c_{m,t} D(p_{m,t}\|q_{m,t}), \qquad \nabla_{z_{m,t}} \mathcal{L}_D = c_{m,t} \nabla_{z_{m,t}} D(p_{m,t}\|q_{m,t}), \tag{19}$$

where $c_{m,t} = 1/N$ for the global token mean, $N = \sum_m |a_m|$, and $c_{m,t} = 1/|a_m|$ for OPAD's sum of per-step means. Thus $\|\nabla_z \mathcal{L}_D\|_\infty \le C \max_{m,t} c_{m,t}$ whenever every position satisfies $\|\nabla_{z_{m,t}} D\|_\infty \le C$.

Part 1: JSD trajectory bound is independent of trajectory length. Lemma 1.2 gives $\|\nabla_{z_{m,t}} \mathrm{JSD}_\beta(p\|q)\|_\infty \le 2$ at every position. Equation 19 therefore gives

$$\left\|\nabla_z \mathcal{L}_{\mathrm{JSD}}(\tau)\right\|_\infty \le 2 \max_{m,t} c_{m,t} \le 2. \tag{20}$$

The global mean sharpens this to $2/N$; OPAD gives at most 2. The result controls the gradient with respect to the stacked logits: one support mismatch cannot create an arbitrarily large local signal. It does not bound $\nabla_\theta \mathcal{L}$, which also depends on the model Jacobians and can accumulate across positions.

Part 2: Reverse KL is unbounded under privileged p–q gaps. By Eq. 3, $D^{\leftarrow}_{\mathrm{KL}}(p\|q) = D_{\mathrm{KL}}(q\|p)$. Direct computation via the softmax Jacobian gives

$$\begin{aligned}\nabla_{z_i} D^{\leftarrow}_{\mathrm{KL}}(p\|q) &= \sum_v \tfrac{\partial q(v)}{\partial z_i}\Big[\log \tfrac{q(v)}{p(v)} + 1\Big] - \underbrace{\sum_v \tfrac{\partial q(v)}{\partial z_i}}_{=0} \\ &= \sum_v q(v)(\delta_{iv} - q(i)) \log \tfrac{q(v)}{p(v)}.\end{aligned} \tag{21}$$

Equivalently,

$$\nabla_{z_i} D^{\leftarrow}_{\mathrm{KL}}(p\|q) = q(i)\left[\log \frac{q(i)}{p(i)} - \mathbb{E}_{v\sim q} \log \frac{q(v)}{p(v)}\right]. \tag{22}$$

Fix q with $q(i) \in (0,1)$. To exhibit the divergence, set $p(i) = \varepsilon$ and $p(j) = (1-\varepsilon)r(j)$ for $j \ne i$, where $r(j) > 0$ and $\sum_{j\ne i} r(j) = 1$. The centered gradient above then gives

$$\frac{\partial D_{\mathrm{KL}}(q\|p)}{\partial z_i} = q(i)(1-q(i)) \log(1/\varepsilon) + O(1) \longrightarrow +\infty \qquad (\varepsilon \downarrow 0).$$

The remaining terms are bounded because q and r are fixed. For any fixed sampled trajectory, the reduction coefficients $1/N$ and $1/|a_m|$ are positive constants, so neither reduction removes this divergence. Thus no finite bound independent of the teacher probabilities exists for $\|\nabla_z \mathcal{L}^{\leftarrow}_{\mathrm{KL}}(\tau)\|_\infty$.

Part 3: Forward KL has bounded logit gradients. Forward KL reduces to the cross-entropy gradient

$$\nabla_{z_i} D^{\rightarrow}_{\mathrm{KL}}(p\|q) = q(i) - p(i) \in [-1, 1], \tag{23}$$

so $\|\nabla_z D^{\rightarrow}_{\mathrm{KL}}(p\|q)\|_\infty \le 1$ uniformly. App. D.5 discusses the forward-KL ablation. □

D.4 Derivation for Proposition 2

At a fixed student-visited prefix, write $q(v)$ for the student distribution and $p_k(v) > 0$ for teacher k's post-debate distribution. The confidence weights satisfy $w_k \ge 0$ and $\sum_k w_k = 1$. Define the unnormalized geometric pool

$$g(v) = \prod_{k=1}^{K} p_k(v)^{w_k}, \qquad Z_{\mathrm{geo}} = \sum_{u\in\mathcal{V}} g(u), \qquad \bar{p}_{\mathrm{geo}}(v) = \frac{g(v)}{Z_{\mathrm{geo}}}. \tag{24}$$

Expanding the reverse-KL objective in Eq. 8 gives

$$\begin{aligned}\sum_{k=1}^{K} w_k D_{\mathrm{KL}}(q\|p_k) &= \sum_{v\in\mathcal{V}} q(v)\left(\log q(v) - \sum_{k=1}^{K} w_k \log p_k(v)\right) \\ &= \sum_{v\in\mathcal{V}} q(v) \log \frac{q(v)}{g(v)} = D_{\mathrm{KL}}(q\|\bar{p}_{\mathrm{geo}}) - \log Z_{\mathrm{geo}}.\end{aligned} \tag{25}$$

The normalizer Z_{geo} depends only on the detached teacher distributions and confidence weights. The objective therefore has the same student gradient and minimizer as $D_{\mathrm{KL}}(q\|\bar{p}_{\mathrm{geo}})$, proving Eq. 6. The

log-odds relation follows directly:

$$\log \frac{\bar{p}_{\text{geo}}(u)}{\bar{p}_{\text{geo}}(v)} = \log \frac{g(u)}{g(v)} = \sum_{k=1}^{K} w_k \log \frac{p_k(u)}{p_k(v)}. \tag{26}$$

The implemented loss computes one reverse-KL term per teacher and then takes their confidence-weighted sum. By the identity above, this objective is equivalent to reverse KL against the normalized weighted geometric mean $\bar{p}_{\text{geo}}$; it does not require a separate aggregation step. At each prefix, the resulting log-odds pooling favors token choices supported across the post-debate teachers. In the 14B+8B→4B ablation, reverse KL achieves the highest pass@1 on both code benchmarks (Table 7).

D.5 Forward KL in the Divergence Ablation

Equation (23) gives $\nabla_{z_i} D_{\text{KL}}^{\rightarrow}(p\|q) = q(i) - p(i) \in [-1, 1]$. Forward KL therefore also has a bounded local gradient. Its teacher-weighted cross-entropy term penalizes low student probability wherever the teacher assigns mass; this is the distributional basis of its mode-covering behavior. In tool use, a teacher can assign probability to several plausible actions, and forward KL encourages the student to retain those alternatives at the current prefix.

This property alone does not determine whether later actions form a successful trajectory: each next-token distribution is conditioned on the realized prefix. In Table 7, forward KL scores below JSD on all three agentic benchmarks. Together with Proposition 1, this comparison shows that bounded logit gradients alone do not determine the task-level ranking. JSD's shared mixture supplies an additional distinction in how disagreement is weighted, motivating its use in the tested agentic setting.

E Case Study: Correcting Tool Arguments through Debate

Complementing the schematic in Figure 1, we present an *illustrative* worked-out example from a different task domain in the ToolACE training set (Liu et al., 2025), where a single 14B teacher fails but a two-round debate between the 14B and 8B teachers yields the correct answer. We show the complete debate trajectory, including each teacher's brief analysis and function-call output, to illustrate how deliberation corrects complementary errors. *For readability, the inline analyses are lightly paraphrased from the model outputs, while the function-call arguments and confidence scores are reproduced verbatim.*

Problem: ToolACE Training-Set Tool-Calling Task

System: Current time is `2020-10-06 04:12:36`. Two functions are available. `travel_budget_planner` requires the destination, dates, accommodation, and activities; `travel_insurance_quote_generator` requires age, countries, trip duration, coverage type, and deductible.

User: "I plan on taking a trip **next month** from the **16th to the 30th**. I'll be visiting Japan and I prefer staying in a hotel. Some of my preferred activities would include sightseeing and local cuisine tasting. Can you help me with a budget plan for this trip? Also, can you get me an insurance quote for this trip? I am 34 with no medical conditions and I'd like an all-inclusive insurance package with a deductible of $200."

Key reasoning challenges: (1) "next month" from Oct 6 = **November** 2020; (2) 16th to 30th = **14 days**; (3) both functions must be called with correctly nested parameters.

Ground Truth

```
[travel_budget_planner(travel_details={"destination": "Japan",
  "dates": {"start_date": "2020-11-16", "end_date": "2020-11-30"},
  "preferences": {"accommodation": "hotel", "activities":
["sightseeing", "local cuisine tasting"]}}]),
 travel_insurance_quote_generator(insurance_details={"personal_info":
{"age": 34},
  "trip_info": {"countries_visited": ["Japan"], "trip_duration": 14},
  "coverage_preferences": {"coverage_type": "all_inclusive",
"deductible": 200}})]
```

Setting 1: Single 14B Teacher ✗ **Incorrect**

Model output (paraphrased):

"The user wants to plan a trip and get an insurance quote. Destination: Japan. Dates: from the 16th to the 30th of next month. Since the current time is 2020-10-06, **the trip dates would be October 16 to October 30**. Accommodation: hotel. Trip duration: Oct 16 to Oct 30 = 14 days."

Function call (key arguments):

```
start_date:     "2020-10-16" ← wrong month
end_date:       "2020-10-30" ← wrong month
trip_duration:  14 ✓
```

Confidence: $c = 90$

Round 1: Expert A (14B Teacher) ✗ **Incorrect**

Model output (paraphrased):

"Dates: next month, 16th to 30th. Current date is October 6, 2020. **So the dates are October 16 to October 30, 2020.** Duration: 16th to 30th = 14 days."

Function call (key arguments):

```
start_date:     "2020-10-16" ← wrong month
end_date:       "2020-10-30" ← wrong month
trip_duration:  14 ✓
```

Confidence: $c = 85$

Round 1: Expert B (8B Teacher) ✗ **Incorrect**

Model output (paraphrased):

"The user wants to visit Japan. They said 'next month from the 16th to the 30th'. Current time is 2020-10-06, **so next month is November**. Start date: 2020-11-16, End date: 2020-11-30. Trip duration: **30 − 16 = 15 days** (counting both endpoints)."

Function call (key arguments):

```
start_date:     "2020-11-16" ✓
end_date:       "2020-11-30" ✓
trip_duration:  15 ← wrong duration
```

Confidence: $c = 80$

Key observation: Errors are **complementary**: Expert A gets the month wrong but the duration right, while Expert B gets the month right but the duration wrong.

Round 2: Expert A (14B Teacher) ✓ **Correct**

Model output (paraphrased, after seeing Expert B's Round 1 answer):

"Looking at Expert B's answer, they used **November** dates while I used October dates. Let me reconsider. The user said 'next month' and the current date is October 6. **'Next month' means the month after October, which is November. Expert B is correct on this point; I made an error.**

However, Expert B calculated trip duration as 15 days. From November 16 to November 30, the duration is **30 − 16 = 14 days**. My original duration of 14 was correct.

Let me revise: use the correct November dates while keeping the correct trip duration of 14 days."

Function call (key arguments):

```
start_date:     "2020-11-16" ✓ corrected
end_date:       "2020-11-30" ✓ corrected
trip_duration:  14 ✓ retained
```

Confidence: $c = 95$ (↑ from 85)

Table 10: **Debate progression.** Green = correct, red = incorrect (also indicated by ✓/× markers for colorblind readers). Complementary Round 1 errors cancel after one debate round; both teachers converge to the ground truth with differentiated confidence.

Setting	Teacher	start_date	end_date	trip_dur.	Conf.
Single teacher	14B	10-16	10-30	14	90
Round 1	14B	10-16	10-30	14	85
	8B	11-16	11-30	15	80
Round 2	14B	11-16	11-30	14	**95**
	8B	11-16	11-30	14	90
Ground truth	–	11-16	11-30	14	–

Student on-policy output (`start_date` argument):

`" start_date ": " 2020 -10 -16 "`

Ground truth: `-11` (November); student writes `-10` (October).

Teacher responses (schematic):

Teacher setting	Highlighted argument
Single 14B	`" start_date ": " 2020 -10 -16 "` *The single teacher repeats the incorrect month.*
MAD-OPD Round 2	`" start_date ": " 2020 -11 -16 "` *Both teachers revise the month to `-11` (November).*

Figure 8: **Schematic comparison of teacher responses** for the student's `start_date` argument. Green marks agreement with the student's output; red highlights the month revised by both teachers after debate. The single 14B teacher repeats the incorrect month, whereas Round 2 responses correct it to November.

Round 2: Expert B (8B Teacher) ✓ **Correct**

Model output (paraphrased, after seeing Expert A's Round 1 answer):

"Expert A used October dates while I used November. The user said 'next month' and the current time is October 6, so next month is November. **I believe my date calculation is correct.**

However, Expert A used `trip_duration` of 14 while I used 15. Let me reconsider. From November 16 to November 30: 30 − 16 = 14 days. **In travel planning, the standard convention is to count the difference between end and start dates. So 14 is the more appropriate value.**"

Function call (key arguments):

`start_date:` `"2020-11-16"` ✓ retained
`end_date:` `"2020-11-30"` ✓ retained
`trip_duration:` `14` ✓ **corrected**

Confidence: $c = 90$ (↑ from 80)

Summary. Table 10 summarizes the revisions. In round 2, the 14B teacher corrects the month and the 8B teacher corrects the duration. Their final confidence scores of 95 and 90 give weights $w_{14B} \approx 0.513$ and $w_{8B} \approx 0.487$ under Eq. 7c.

Correction through debate. Figure 8 illustrates the month error in the student's `start_date` argument and the teachers' revisions. The single 14B teacher repeats the student's incorrect month, `-10`. After Round 2, both teachers use the correct month, `-11`. Their revised responses form part of the privileged context used to supervise the student's sampled action.

Implications for distillation. The Round 2 debate changes the teacher distributions used for privileged force-decoding. The student therefore receives token-level supervision shaped by error

detection and revision, even though the debate transcript remains hidden from the student. In this example, the revised consensus corrects an error shared by the student and the stronger teacher, illustrating the advantage of MAD-OPD over single-teacher OPD.

F Debate Prompt Templates

This appendix lists the prompt templates for the two debate rounds. Round 1 elicits independent responses. Round 2 presents both responses for revision. We instantiate separate templates for agentic tasks and code generation; placeholders appear in `{braces}`. Confidence weighting appends the suffix in §F.3 to every prompt.

F.1 Round 1: Independent Response

Each teacher receives the current state s_m and generates an independent response. For code generation, $s_m = x$; for tool use, it also contains the interaction history. The system instruction is domain-specific:

Round 1: Agentic Tasks

```
You are an expert in AI tool-use and function calling.  Please analyze
the following user request and provide the best tool-use response.

User request:  {prompt}

Your response:
```

Round 1: Code Generation

```
You are an expert programmer specializing in competitive programming
and code golf.  Please solve the following programming challenge.
Focus on algorithm correctness, edge case handling, and code
efficiency.

Problem:
{prompt}

Your code solution:
```

F.2 Round 2: Debate with Shared History

Each teacher receives the current state together with both Round 1 responses, and is asked to revise its answer after considering the other expert's perspective. The prompt allows each teacher to retain its answer if it disagrees with the other teacher.

Round 2: Agentic Tasks, Expert A

```
{prompt}

Two experts have provided their tool-use responses to this request:

[Expert A, Round 1]:
{teacher_1_answer}

[Expert B, Round 1]:
{teacher_2_answer}

You are Expert A. You don't have to agree with the other expert's
response.  Please continue the debate about the best tool-use strategy.
Having seen the other expert's response, please reconsider and provide
your improved response focusing on tool selection accuracy, parameter
correctness, and call sequence optimization.
```

Round 2: Code Generation, Expert A

```
{prompt}

Two expert programmers have provided their code solutions to this
challenge:

[Expert A, Round 1]:
{teacher_1_answer}

[Expert B, Round 1]:
{teacher_2_answer}

You are Expert A. You don't have to agree with the other expert's
solution.  Please continue the debate about the best approach.  Having
seen the other expert's code, please reconsider and provide your
improved solution focusing on algorithm correctness, edge case
handling, time/space complexity, and code conciseness.
```

F.3 Confidence Scoring Suffix

With confidence weighting, the following line is appended to every prompt in both rounds. The final-round scores c_k are normalized into teacher weights w_k by Eq. 7c.

Confidence suffix

```
End your response with a confidence score line:  Confidence Score:  N
(where N is 0-100)
```

After the debate, the $K \cdot R = 4$ responses form the privileged transcript $\mathcal{H}_m^R$ used during force-decoding in §4.3:

Debate transcript structure

```
[Debate Start]
[Expert A Initial Answer]:  {round1_teacher1}
[Expert B Initial Answer]:  {round1_teacher2}
[Expert A Revised Answer]:  {round2_teacher1}
[Expert B Revised Answer]:  {round2_teacher2}
[Debate End]
```

G Broader Impact

MAD-OPD transfers capabilities expressed by an open-weight teacher pool into a smaller student. This can reduce inference cost for agentic tool use and code generation while making the distilled capabilities cheaper to deploy at scale. We did not evaluate safety or refusal behavior, so the student should not be assumed to preserve the teachers' safeguards. Any checkpoint release should retain the base-model license, document the training setup and intended use, and apply safeguards appropriate to the deployment context.