METAOPD: META-LEARNED TOKEN WEIGHTING FOR ON-POLICY DISTILLATION

Anonymous authors
Paper under double-blind review

ABSTRACT

On-policy distillation (OPD) trains a student on its own generated responses using token-level teacher supervision. However, uniform weighting overlooks differences in token learning value, while existing weighting methods rely on predefined mappings from prediction signals to token weights. These mappings are not learned from the effectiveness of the resulting student updates, limiting their ability to adapt to evolving learning needs. In this paper, we propose **MetaOPD**, a bilevel optimization framework that jointly learns the student model and a lightweight token-weighting network. The inner objective updates the student through weighted OPD, while the outer objective optimizes the weighting network using validation loss on reference solutions after a virtual student update. Differentiating through this update connects weighting decisions to their effects on post-update performance, allowing the mapping from prediction signals to token weights to evolve alongside the student. Experiments on six mathematical reasoning and three out-of-domain datasets, covering two student scales and seven baselines, demonstrate the effectiveness of MetaOPD, with Avg@8/Pass@8 gains over OPD of 1.99/5.97 percentage points for the 0.6B student and 2.25/6.41 points for the 1.7B student.

1 INTRODUCTION

Off-policy distillation trains a student on teacher-generated sequences, which can lead to a distribution mismatch between training and inference (Lin et al., 2020; Agarwal et al., 2024). To mitigate this issue, on-policy distillation (OPD) trains the student on its own generated trajectories, with a frozen teacher providing dense token-level supervision. OPD typically employs a mode-seeking reverse-KL objective that encourages the student to capture the teacher's high-probability behaviors (Wen et al., 2023; Gu et al., 2024). However, experiments by (Li et al., 2026b) show that tokens differ substantially in their learning value, a distinction that standard OPD's uniform weighting fails to capture.

Recent approaches address this limitation by allocating token-level weights through **static proxy rules**. For example, EOPD (Jin et al., 2026) uses predictive entropy, while TIP (Xu et al., 2026b) leverages teacher–student disagreement. These methods provide useful signals for allocating supervision based on predictive uncertainty or differences between model predictions. However, their mappings from proxy signals to token weights are predefined rather than adapted to the student's actual learning gains. As training progresses, the student's capabilities and generated contexts evolve, and the learning benefit associated with the same proxy signal may change accordingly. Thus, even when token weights vary across inputs, a fixed weighting rule may not transfer effectively across training stages or task contexts. *The central challenge is to learn a weighting rule that adapts supervision to the student's evolving learning needs through feedback from the resulting updates.*

To examine whether different token weighting strategies translate into actual learning gains, we conduct a motivating experiment. As illustrated in Figure 1(a), we compare assigning weights through predefined proxies with learning weights. Figure 1(b) reports validation loss reductions relative to Uniform OPD, with larger values indicating greater gains. EOPD yields only a modest improvement, while TIP and random weighting perform close to the uniform baseline. In contrast, learning weights through validation-loss feedback produces a substantially larger improvement. These results suggest that, in this setting, predefined rules do not fully exploit the potential benefits of token weighting, whereas the student's update performance provides valuable feedback for weight allocation. The key is therefore not only to distinguish tokens using prediction signals, but also to

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.

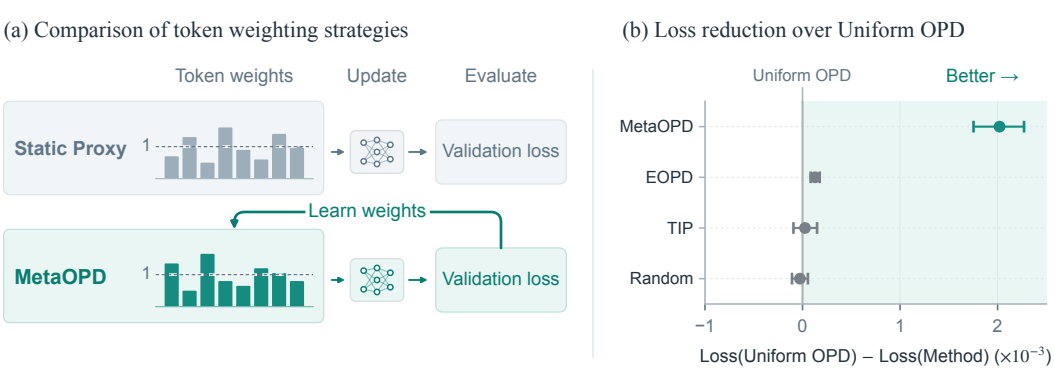


Figure 1: **Motivating experiment on token weighting strategies.** (a) Static proxy methods assign token weights using predefined signals, while MetaOPD learns weights through validation-loss feedback. (b) Validation loss reductions relative to Uniform OPD for MetaOPD, EOPD (Jin et al., 2026), TIP (Xu et al., 2026b), and random token weighting. MetaOPD achieves the largest reduction, supporting the benefit of learning token weights aligned with the validation objective.

adapt the signal-to-weight mapping based on the effectiveness of the resulting updates. This leads to our central question: **How can we use the student's post-update performance to learn a token weighting rule that translates prediction signals into effective supervision?**

In this paper, we take a step forward by proposing **MetaOPD**, an on-policy distillation framework that jointly optimizes the student model and the token-weighting rule. Specifically, we introduce a lightweight token-weighting network that maps prediction signals to token weights. The student and the weighting network perform two coupled learning tasks: *the student improves through weighted teacher supervision*, while *the weighting network learns to allocate supervision based on the effectiveness of student updates*. To achieve this, we formulate their joint learning as a bilevel optimization problem. At the inner level, we update the student on its own generated responses using an OPD loss weighted by the network's outputs. At the outer level, we evaluate the student's loss on validation reference solutions after a virtual update and use this loss to optimize the weighting network. Differentiating through the virtual update allows the student's post-update performance to guide token weight allocation, enabling the network to learn which allocations lead to greater subsequent improvements. Alternating between the two levels allows the student and the weighting rule to evolve together, replacing predefined proxy weighting with adaptive supervision allocation driven by update outcomes. Our main contributions are summarized as follows:

- We identify a gap between static proxy weighting and actual learning gains in OPD, and empirically motivate learning token weights from post-update performance feedback.
- We propose MetaOPD, a bilevel framework that jointly optimizes the student through weighted OPD and a lightweight token-weighting network through update validation loss.
- We conduct extensive experiments across six reasoning and three out-of-domain datasets on two student scales against seven baselines, demonstrating the effectiveness of our MetaOPD.

2 Related Work

On-policy distillation (OPD). On-policy distillation (OPD) trains a student to match teacher predictions on student-generated trajectories (Agarwal et al., 2024). Compared with distillation on fixed data or teacher-generated sequences (Hinton et al., 2015; Kim & Rush, 2016), it reduces training–inference mismatch by providing teacher feedback at prefixes the student actually visits. One line of work improves OPD through the design of distillation objectives and optimization strategies (Gu et al., 2024; Ko et al., 2024; Yang et al., 2026; Yu et al., 2026). Another focuses on selecting training prompts, improving rollout trajectories, and refining teacher supervision (Li et al., 2026b; Xu et al., 2026a; Qu et al., 2026). Since tokens differ in learning value, assigning different weights to tokens within the OPD loss offers another way to improve OPD.

Static proxy weighting OPD. Static proxy weighting OPD methods start from the observation that dense teacher feedback is not equally useful at every token. Existing approaches use signals

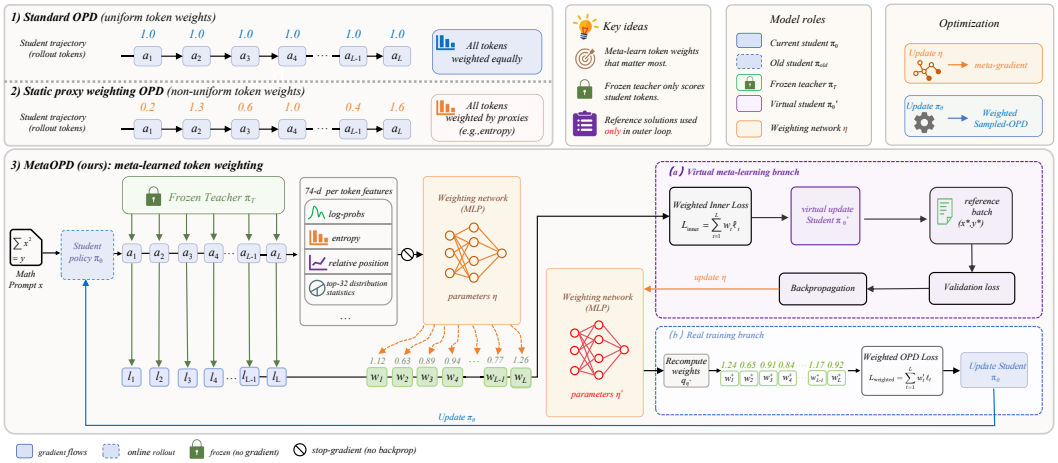


Figure 2: **Overview of MetaOPD.** (1) Standard OPD uses uniform token weights, while (2) static proxy weighting OPD assigns token weights through predefined rules based on proxy signals. (3) MetaOPD uses two branches: (a) the weighting network learns from the student's validation loss after a virtual weighted OPD update; (b) the updated network recomputes weights for the actual student update. The schematic sums omit valid-token normalization, and weights in the actual-update branch refer to those recomputed after the weighting-network update.

such as entropy and teacher–student disagreement to identify informative tokens and emphasize their supervision through token selection, loss adjustment, or reweighting (Jin et al., 2026; Ko et al., 2026; Xu et al., 2026b; Wang et al., 2026; Li et al., 2026c). Another line accounts for changes in supervision reliability along a trajectory, using accumulated prefix discrepancy or token position to guide weighting (Xie et al., 2026; Liu et al., 2026). However, these methods use predefined mappings from proxy signals to supervision allocation, without learning from post-update feedback how those mappings should adapt to the student's evolving learning needs.

Bilevel Optimization. Bilevel optimization provides a framework for learning example or token weights using feedback on the updated model. Early meta-reweighting methods used this structure to learn example importance under class imbalance and label noise (Ren et al., 2018; Shu et al., 2019). Recent studies have extended this principle to language-model training, using outer-loop feedback to replace fixed weighting rules with update-aware supervision (Hu et al., 2023; Pan et al., 2025; Yu et al., 2025; Zhang et al., 2025; Li et al., 2026a). Across these settings, a common view emerges: the value of a training signal is not fixed, but depends on the model update it induces and the reference objective used to assess that update. MetaOPD applies this principle to token weighting within the OPD loss on student-generated trajectories.

3 Methodology

This section first introduces the token-weighted on-policy distillation objective (§3.1), then presents the bilevel optimization procedure of MetaOPD (§3.2), and finally explains the role of individual tokens in the weighting-network update and describes the gradient computation (§3.3).

3.1 Token-Weighted On-Policy Distillation

Let $\mathcal{D}_{\text{train}}$ denote the training distribution over question–solution pairs. We write $\mathcal{D}^X_{\text{train}}$ for the corresponding distribution of questions. The student π_θ has trainable parameters θ, and π_{T} is the frozen teacher. At the start of each rollout round, we set $\theta_{\text{old}} \leftarrow \theta$. For $x_i \sim \mathcal{D}^X_{\text{train}}$, we sample a response from $\pi_{\theta_{\text{old}}}$:

$$y_i = (a_{i,1}, \ldots, a_{i,T_i}) \sim \pi_{\theta_{\text{old}}}(\cdot \mid x_i), \qquad s_{i,t} = (x_i, a_{i,<t}), \tag{1}$$

where $s_{i,t}$ is the visited prefix and $a_{i,t}$ the sampled token. The mask $m_{i,t} \in \{0, 1\}$ marks valid completion positions, including an observed end-of-sequence token.

For each sampled token, write $u^{\theta}_{i,t} = \log \pi_\theta(a_{i,t} \mid s_{i,t})$, with $u^{\text{old}}_{i,t}$ denoting the rollout-time log-probability and $u^{\text{T}}_{i,t}$ the teacher log-probability after aligning token identities. All three use full-vocabulary normalization; Appendix A.1 specifies the teacher-view mapping. The detached teacher-minus-rollout log-probability difference and the current-to-old probability ratio are

$$A_{i,t} = \text{sg}[u^{\text{T}}_{i,t} - u^{\text{old}}_{i,t}], \qquad r_{i,t}(\theta) = \exp(u^{\theta}_{i,t} - u^{\text{old}}_{i,t}), \tag{2}$$

where sg preserves its argument's value but blocks gradients through it. Following the sampled reverse-KL update used in Relay-OPD (Xu et al., 2026a), let $r^{\text{clip}}_{i,t} = \text{clip}(r_{i,t}, 1-\epsilon_l, 1+\epsilon_h)$, with clipping margins ϵ_l, ϵ_h, and define the per-token OPD term as

$$\ell^{\text{OPD}}_{i,t}(\theta) = \max\Big(-A_{i,t} r_{i,t}(\theta), -A_{i,t} r^{\text{clip}}_{i,t}\Big). \tag{3}$$

At $\theta = \theta_{\text{old}}$, the unclipped form gives an unbiased single-action estimate of the student-to-teacher reverse-KL gradient at the fixed visited prefix, using the induced teacher distribution defined in Appendix A.1. Away from that policy, the cached log-probability difference and clipping define a local surrogate.

A shared weighting network q_η, with trainable parameters η, maps each token's descriptor $f_{i,t} \in \mathbb{R}^{74}$ to a scalar score. The descriptor contains student and teacher log-probabilities of the sampled token, entropy, prediction agreement, position, and ranked top-32 log-probability profiles (Table 4). These profiles are inputs only and do not restrict the OPD loss vocabulary. The feature $\delta_{i,t} = u^{\text{T}}_{i,t} - u^{\theta}_{i,t}$ uses current-student log-probabilities, whereas $A_{i,t}$ uses cached rollout-policy log-probabilities. All numeric inputs are detached; descriptors are constructed without reference solutions.

We convert the network scores into positive weights with unit mean within each response. For $n_i = \sum_t m_{i,t} > 0$, define

$$\begin{aligned} z_{i,t} &= q_\eta(\text{sg}[f_{i,t}]), & \bar{z}_i &= \frac{1}{n_i}\sum_s m_{i,s} z_{i,s}, \\ \widetilde{w}_{\eta,i,t} &= 1 + \rho \tanh(z_{i,t} - \bar{z}_i), & w_{\eta,i,t} &= m_{i,t}\widetilde{w}_{\eta,i,t}\frac{n_i}{\sum_s m_{i,s}\widetilde{w}_{\eta,i,s}}. \end{aligned} \tag{4}$$

Centering expresses each token's score relative to the mean over valid tokens in the same response. Normalization then keeps the mean weight over these tokens at one, allowing the network to adjust their relative contributions within the OPD loss. Both operations are differentiated with respect to η. We use residual scale $\rho = 0.25$ and a zero-initialized output head, so all valid weights start at one. Completely masked responses receive zero weights. The architecture, weight bounds, and numerical safeguards appear in Appendices A.2 and B.

For an optimizer minibatch B, the weighted OPD objective is

$$\mathcal{L}_{\text{in}}(\theta, \eta; B) = \frac{1}{N_B}\sum_{i,t} m_{i,t} w_{\eta,i,t} \ell^{\text{OPD}}_{i,t}(\theta), \qquad N_B = \sum_{i,t} m_{i,t}. \tag{5}$$

The numerator and valid-token count are reduced over the complete optimizer minibatch. Setting all valid weights to one recovers uniform sampled OPD.

3.2 Bilevel Optimization of MetaOPD

Figure 2 illustrates the bilevel procedure: the student learns from weighted OPD, while the weighting network learns from the student's validation loss after a virtual student update. Here validation loss is the teacher-forced token loss on training reference solutions, used as the outer objective defined below. Following gradient-based meta-reweighting and Meta-Weight-Net (Ren et al., 2018; Shu et al., 2019), each optimizer minibatch involves a virtual student update, a weighting-network update, and an actual student update.

Virtual student update. Teacher statistics are cached with the rollout. At the start of each optimizer minibatch, we compute current-student statistics and detach the descriptors used to obtain the token weights. Let S collect the incoming student AdamW moments, step counters, and parameter-group hyperparameters. Write $\boldsymbol{g}_{\text{in}}(\eta) := \nabla_\theta \mathcal{L}_{\text{in}}(\theta, \eta; B)$ for the student gradient, whose dependence on η is through the token weights. One virtual step defines the virtual student parameters

$$\theta'(\eta) = \mathcal{A}_{\text{AdamW}}(\theta, \mathcal{C}(\boldsymbol{g}_{\text{in}}(\eta)); \mathsf{S}). \tag{6}$$

The virtual and actual student steps use the same AdamW update rule (Loshchilov & Hutter, 2017). Here $\mathcal{C}$ applies full-model global-norm clipping, and $\mathcal{A}_{\mathrm{AdamW}}$ reads the current student optimizer state and scheduled learning rates. Differentiation with respect to η holds θ, S, the rollout, and detached descriptors fixed. The dependence through the token weights and student gradient is retained, including the new virtual moments, clipping coefficient, and AdamW parameter map. This operation changes neither the real parameters nor the stored moments, step counters, or scheduler. Appendix A.3 expands the two maps.

Weighting-network update. Draw a batch $B^{\mathrm{ref}} = \{(x_j^\star, y_j^\star)\}$ of prompts and reference solutions from a separately shuffled stream over $\mathcal{D}_{\mathrm{train}}$; overlap with the rollout stream is allowed. At $\theta'(\eta)$, the student predicts each solution token given the prompt and preceding solution tokens. We compute validation loss under teacher forcing, averaging over valid target tokens:

$$\mathcal{L}_{\mathrm{out}}(\theta'; B^{\mathrm{ref}}) = -\frac{1}{N_{\mathrm{ref}}} \sum_{j,t} m_{j,t}^{\mathrm{ref}} \log \pi_{\theta'}(y_{j,t}^\star \mid x_j^\star, y_{j,<t}^\star), \qquad N_{\mathrm{ref}} = \sum_{j,t} m_{j,t}^{\mathrm{ref}}. \tag{7}$$

The mask $m_{j,t}^{\mathrm{ref}}$ selects valid reference-solution tokens; N_{ref} is counted separately from N_B. To regularize departures from uniform weighting, define

$$\mathcal{R}_w(\eta; B) = \frac{1}{N_B} \sum_{i,t} m_{i,t} (w_{\eta,i,t} - 1)^2. \tag{8}$$

At global student step r, let $\lambda_w^{(r)}$ be the weight-regularization coefficient, fixed for this step. We optimize the one-step meta-objective

$$\mathcal{J}_{\mathrm{meta}}(\eta) = \mathcal{L}_{\mathrm{out}}(\theta'(\eta); B^{\mathrm{ref}}) + \lambda_w^{(r)} \mathcal{R}_w(\eta; B). \tag{9}$$

Applying the chain rule to the meta-objective gives the *meta-gradient* with respect to η:

$$\nabla_\eta \mathcal{J}_{\mathrm{meta}} = \left(\frac{\partial \theta'(\eta)}{\partial \eta}\right)^\top \nabla_{\theta'} \mathcal{L}_{\mathrm{out}}\big(\theta'(\eta); B^{\mathrm{ref}}\big) + \lambda_w^{(r)} \nabla_\eta \mathcal{R}_w(\eta; B). \tag{10}$$

The first term propagates the validation-loss gradient through the virtual update; the second comes from the weight regularizer. Let M denote the weighting network's AdamW state and $\mathrm{AdamWStep}$ an update returning parameters and state. One update per student minibatch gives

$$(\eta^+, \mathsf{M}^+) = \mathrm{AdamWStep}_\eta(\eta, \nabla_\eta \mathcal{J}_{\mathrm{meta}}; \mathsf{M}). \tag{11}$$

Section 3.3 expands this gradient into token contributions and describes its computation. The weighting-network learning rate and $\lambda_w^{(r)}$ follow the student learning-rate schedule in Appendix B.4.

Actual student update. We discard the virtual parameters and recompute token weights with q_{η^+} on the same detached descriptors. The actual student gradient uses these weights as constants:

$$\begin{aligned} \boldsymbol{g}_{\mathrm{real}} &= \frac{1}{N_B} \sum_{i,t} m_{i,t}\, \mathrm{sg}[w_{\eta^+,i,t}] \nabla_\theta \ell_{i,t}^{\mathrm{OPD}}(\theta), \\ (\theta^+, \mathsf{S}^+) &= \mathrm{AdamWStep}_\theta(\theta, \mathcal{C}(\boldsymbol{g}_{\mathrm{real}}); \mathsf{S}). \end{aligned} \tag{12}$$

This step starts from the original (θ, S), not from θ', and advances the real optimizer state. Its objective contains only weighted OPD supervision on student-generated responses; validation loss trains the weighting network through the meta-objective.

Algorithm 1 One MetaOPD rollout round

Require: Student parameters and AdamW state (θ, S); weighting parameters and meta-AdamW state (η, M); frozen teacher π_{T}
Require: Separately shuffled reference-solution stream $\mathcal{G}$; minibatch count K; global student-step index r
Ensure: Updated (θ, S), (η, M), and r
1: Set $\theta_{\mathrm{old}} \leftarrow \theta$; sample prompts from $\mathcal{D}^X_{\mathrm{train}}$ and generate student responses.
2: Query the teacher at visited prefixes; cache teacher statistics, old-policy log-probabilities, and masks.
3: Partition the rollout into K optimizer minibatches.
4: **for** each minibatch B **do**
5: Draw $B^{\mathrm{ref}} \leftarrow \mathrm{Next}(\mathcal{G})$; compute and detach current-student descriptors.
6: Compute token weights and $\mathcal{L}_{\mathrm{in}}$ using Equations (4) and (5).
7: **Virtual update:** construct $\theta'(\eta)$ using Equation (6), leaving (θ, S) unchanged.
8: **Meta-update:** compute $\nabla_\eta \mathcal{J}_{\mathrm{meta}}$ using Equation (10); obtain (η^+, M^+) using Equation (11).
9: Discard θ'; recompute weights with q_{η^+} on the same descriptors and detach them.
10: **Student update:** obtain (θ^+, S^+) from the original (θ, S) using Equation (12).
11: Set $(\theta, \mathsf{S}) \leftarrow (\theta^+, \mathsf{S}^+)$, $(\eta, \mathsf{M}) \leftarrow (\eta^+, \mathsf{M}^+)$, and $r \leftarrow r + 1$.
12: **end for**

3.3 Token-Level Interpretation and Gradient Computation

We examine how a small change in a token's weight within the OPD loss affects validation loss after the virtual student update. Throughout, $\nabla_\eta \mathcal{L}_{\mathrm{out}}$ denotes the gradient with respect to η obtained by differentiating $\mathcal{L}_{\mathrm{out}}(\theta'(\eta); B^{\mathrm{ref}})$ through the virtual update, with the inputs fixed as in §3.2. Define the token gradient $\boldsymbol{g}_{i,t} = \nabla_\theta \ell^{\mathrm{OPD}}_{i,t}(\theta)$. The detached descriptors make weights constant with respect to θ, while preserving their dependence on η, so

$$\boldsymbol{g}_{\mathrm{in}}(\eta) = \frac{1}{N_B} \sum_{i,t} m_{i,t} w_{\eta,i,t} \boldsymbol{g}_{i,t}. \tag{13}$$

Let $\boldsymbol{g}_{\mathrm{out}} = \nabla_{\theta'} \mathcal{L}_{\mathrm{out}}(\theta'; B^{\mathrm{ref}})$, evaluated at $\theta' = \theta'(\eta)$. Write $J_{\mathcal{C}}$ for the clipping Jacobian at $\boldsymbol{g}_{\mathrm{in}}(\eta)$, and $J_{\mathcal{A}}$ for the AdamW parameter-map Jacobian with respect to its gradient input, evaluated at $\mathcal{C}(\boldsymbol{g}_{\mathrm{in}}(\eta))$. Propagating the outer gradient through these maps defines

$$\boldsymbol{d} = -J_{\mathcal{C}}^\top J_{\mathcal{A}}^\top \boldsymbol{g}_{\mathrm{out}}. \tag{14}$$

This derivative-based vector, rather than the actual student parameter change, determines each token's contribution to the weighting-network gradient:

$$c_{i,t} = \langle \boldsymbol{d}, \boldsymbol{g}_{i,t} \rangle, \qquad \nabla_\eta \mathcal{L}_{\mathrm{out}} = -\frac{1}{N_B} \sum_{i,t} m_{i,t} c_{i,t} \nabla_\eta w_{\eta,i,t}. \tag{15}$$

For a valid token, $\partial \mathcal{L}_{\mathrm{out}} / \partial w_{i,t} = -c_{i,t}/N_B$ when the other normalized-weight coordinates are held fixed. Thus, $c_{i,t} > 0$ means that a small increase in this weight alone locally reduces validation loss. Learning rates are already included in $J_{\mathcal{A}}$; adding $\lambda_w^{(r)} \nabla_\eta \mathcal{R}_w$ to Equation (15) gives the complete meta-gradient in Equation (10). Appendix A.4 provides the intermediate Jacobian and chain-rule steps.

Response normalization couples the weight coordinates. Suppressing η for clarity, let $\overline{w}_i = n_i^{-1} \sum_t m_{i,t} \widetilde{w}_{i,t}$. Differentiating $w_{i,t} = \widetilde{w}_{i,t} / \overline{w}_i$ on valid positions gives

$$\frac{\partial \mathcal{L}_{\mathrm{out}}}{\partial \widetilde{w}_{i,j}} = -\frac{c_{i,j} - \bar{c}_i^{\,w}}{N_B \overline{w}_i}, \qquad \bar{c}_i^{\,w} = \frac{1}{n_i} \sum_t m_{i,t} w_{i,t} c_{i,t}. \tag{16}$$

The gradient with respect to a raw weight therefore depends on $c_{i,j} - \bar{c}_i^{\,w}$, the difference from the response's weighted mean. Score centering, the shared network, and regularization further affect the update, so $c_{i,j} > 0$ need not lead to an increase in the learned weight. This derivative describes a

Table 1: Mathematical-reasoning results (%) with a Qwen3-8B teacher. Avg. is the unweighted average over six benchmarks; benchmark and Avg. scores are rounded separately. Abs. Δ_{OPD} reports MetaOPD–OPD in percentage points. Rel. Δ_{OPD} reports the same difference as a percentage of the OPD score. Delta values are calculated from the displayed scores. Bold marks the best result in each student setting, including ties.

Method	Avg@8							Pass@8						
	MATH500	AMC23	Minerva	HMMT	AIME24	AIME25	**Avg.**	MATH500	AMC23	Minerva	HMMT	AIME24	AIME25	**Avg.**
TEACHER Qwen3-8B ⇝ STUDENT Qwen3-0.6B-Base														
KD	47.90	23.75	14.25	0.42	2.50	0.83	14.94	69.20	50.00	31.25	3.33	6.67	6.67	27.85
GRPO	**53.20**	28.44	16.31	0.83	**4.17**	1.25	17.37	75.00	57.50	32.35	6.67	6.67	10.00	31.36
OPD	50.28	25.00	15.95	0.42	2.92	1.25	15.97	73.00	55.00	31.62	3.33	10.00	6.67	29.94
EOPD	50.40	27.50	16.08	1.67	3.75	1.25	16.78	75.60	52.50	34.19	6.67	13.33	6.67	31.49
FiRe-OPD	50.98	29.38	16.27	0.83	2.92	0.83	16.87	73.60	**62.50**	31.99	3.33	10.00	6.67	31.35
REOPOLD	50.73	26.56	15.35	1.67	3.33	0.00	16.27	74.40	57.50	31.25	3.33	13.33	0.00	29.97
TIP	51.68	29.69	16.45	1.25	2.08	1.25	17.07	74.60	**62.50**	31.99	6.67	10.00	10.00	32.63
MetaOPD	52.88	**30.31**	**16.68**	**2.08**	3.75	**2.08**	**17.96**	**78.40**	**62.50**	**34.56**	**10.00**	**16.67**	**13.33**	**35.91**
Abs. Δ_{OPD}	+2.60	+5.31	+0.73	+1.66	+0.83	+0.83	+1.99	+5.40	+7.50	+2.94	+6.67	+6.67	+6.66	+5.97
Rel. Δ_{OPD}	+5.17%	+21.24%	+4.58%	+395.24%	+28.42%	+66.40%	+12.46%	+7.40%	+13.64%	+9.30%	+200.30%	+66.70%	+99.85%	+19.94%
TEACHER Qwen3-8B ⇝ STUDENT Qwen3-1.7B-Base														
KD	62.93	38.13	27.02	1.25	10.00	3.75	23.85	84.80	72.50	43.38	3.33	20.00	16.67	40.11
GRPO	66.85	37.81	28.26	1.67	8.75	5.42	24.79	83.20	72.50	48.16	3.33	23.33	16.67	41.20
OPD	67.23	38.75	27.80	1.67	8.33	6.67	25.07	84.20	67.50	47.43	6.67	20.00	16.67	40.41
EOPD	67.60	40.63	27.48	1.25	10.83	7.08	25.81	83.80	72.50	44.12	6.67	30.00	20.00	42.85
FiRe-OPD	68.55	40.31	28.45	**2.08**	7.92	7.08	25.73	85.80	72.50	43.38	6.67	16.67	23.33	41.39
REOPOLD	68.85	36.25	**28.77**	1.67	10.83	6.67	25.51	86.40	65.00	44.85	6.67	23.33	20.00	41.04
TIP	68.75	41.25	27.53	1.67	11.25	6.67	26.19	86.80	72.50	45.59	**10.00**	30.00	**26.67**	45.26
MetaOPD	**69.95**	**43.75**	27.71	**2.08**	**12.08**	**8.33**	**27.32**	**88.20**	**77.50**	**48.53**	**10.00**	**33.33**	23.33	**46.82**
Abs. Δ_{OPD}	+2.72	+5.00	-0.09	+0.41	+3.75	+1.66	+2.25	+4.00	+10.00	+1.10	+3.33	+13.33	+6.66	+6.41
Rel. Δ_{OPD}	+4.05%	+12.90%	-0.32%	+24.55%	+45.02%	+24.89%	+8.97%	+4.75%	+14.81%	+2.32%	+49.93%	+66.65%	+39.95%	+15.86%

local effect; it does not guarantee finite-step or long-term improvement. Appendix A.2 expands the normalization derivative.

Reverse-mode vector–Jacobian products (VJPs) first compute $\boldsymbol{g}_{\mathrm{out}}$, then propagate it through AdamW and clipping to obtain $\boldsymbol{d}$. A Jacobian–vector product (JVP) of the vector of per-token OPD terms at the current θ, in direction $\boldsymbol{d}$, gives all $c_{i,t}$ values without materializing the token-by-parameter Jacobian. Recomputing weights with their η-dependence intact, we differentiate the following objective while treating $c_{i,t}$ as constants:

$$\widehat{\mathcal{L}}^{(r)}_{\mathrm{meta}}(\eta; B, \boldsymbol{c}) = -\frac{1}{N_B} \sum_{i,t} m_{i,t} w_{\eta,i,t}\, \mathrm{sg}[c_{i,t}] + \lambda_w^{(r)} \mathcal{R}_w(\eta; B). \tag{17}$$

Holding $c_{i,t}$ fixed retains the Jacobian-product values required by the chain rule without differentiating them again. The resulting η-gradient equals the complete gradient in Equation (10) at the iterate where these values were computed; this is a first-derivative equality at that point, not a global identity between objectives. Appendix A.4 verifies the equality.

4 Experiments

4.1 Experimental Setup

Datasets. We evaluate on six mathematical reasoning benchmarks, *i.e.*, MATH500 (Hendrycks et al., 2021; Lightman et al., 2024), AMC23 (Yang et al., 2024), Minerva Math (Lewkowycz et al., 2022), HMMT 2025 (Balunović et al., 2025), AIME24 (Zhang & Math-AI, 2024), and AIME25 (Zhang & Math-AI, 2025), and assess out-of-domain on three additional datasets, *i.e.*, GPQA-Diamond (Rein et al., 2023), MMLU Pro (Wang et al., 2024), and AlpacaEval 2.0 (Dubois et al., 2024).

Baseline methods. We compare our MetaOPD with seven baselines spanning different learning strategies: KD (Hinton et al., 2015), GRPO (Shao et al., 2024), OPD (Agarwal et al., 2024), EOPD (Jin et al., 2026), FiRe-OPD (Li et al., 2026c), REOPOLD (Ko et al., 2026), and TIP (Xu et al., 2026b).

Implementation details. We use Qwen3 (Yang et al., 2025) models, with Qwen3-8B as the frozen teacher and Qwen3-0.6B-Base and Qwen3-1.7B-Base as students. For all methods, both student models are trained for three epochs on MATH (Hendrycks et al., 2021). MetaOPD samples rollout

Table 2: Out-of-domain results for Qwen3-1.7B-Base after MATH training, with Qwen3-8B as the teacher for distillation methods. Base Model is the student before training. Scores are percentages; higher is better. GPQA-Diamond reports Avg@8/Pass@8, MMLU-Pro reports Pass@1, and AlpacaEval 2.0 reports length-controlled win rate (LC-WR) and win rate (WR). Avg. is the unweighted average of these five metric scores. Bold and underline mark the best and second-best results, respectively; the MetaOPD column is shaded.

Benchmark	Metric	Base Model	KD	GRPO	OPD	EOPD	FiRe-OPD	REOPOLD	TIP	**MetaOPD**
GPQA-Diamond	Avg@8	15.09	21.09	19.26	26.20	25.44	25.82	26.52	<u>27.15</u>	**27.27**
	Pass@8	69.19	75.25	77.27	<u>77.78</u>	76.26	76.77	76.26	74.75	**78.28**
MMLU-Pro	Pass@1	7.28	38.13	41.86	41.57	40.93	41.51	39.52	<u>41.90</u>	**42.30**
AlpacaEval 2.0	LC-WR	1.21	14.72	13.12	15.96	15.19	15.14	<u>16.06</u>	**16.70**	15.98
	WR	2.03	16.09	14.63	17.97	17.17	17.13	17.37	**19.14**	<u>18.08</u>
All metrics	Avg.	18.96	33.06	33.23	35.90	35.00	35.27	35.15	<u>35.93</u>	**36.38**

prompts and reference solutions from independently shuffled versions of the training split, allowing overlap between the two streams. Evaluation examples are excluded from both student and weighting network updates. All methods are evaluated using the same decoding settings and a Math-Verify-based answer verifier. We generate eight responses per problem and report Avg@8, the average accuracy across these responses, and Pass@8, the proportion of problems with at least one correct response. We aggregate each metric using an unweighted average across benchmarks. Full training and evaluation details are provided in Appendix B.

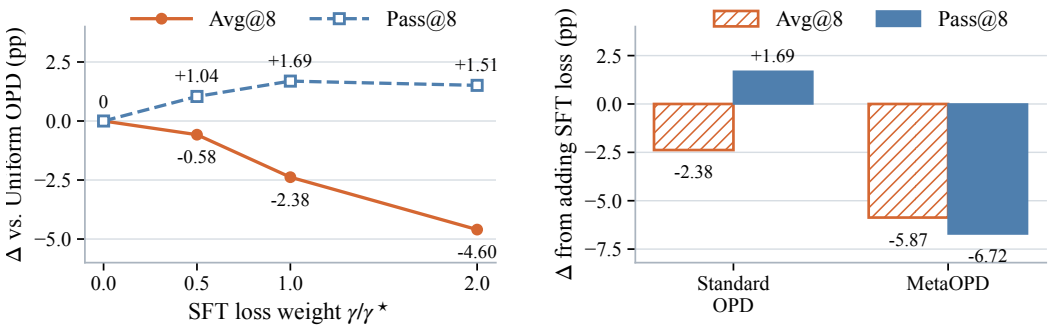
4.2 Main Results

Mathematical-reasoning performance. Table 1 reports the performance of MetaOPD and seven baselines on six mathematical-reasoning benchmarks, using Qwen3-0.6B-Base and Qwen3-1.7B-Base as students. MetaOPD achieves the highest six-benchmark averages for both Avg@8 and Pass@8 at both student scales, including comparisons with adaptive OPD baselines. With the 1.7B student, MetaOPD improves Avg@8/Pass@8 over standard OPD by 2.25/6.41 percentage points and over TIP, which attains the highest aggregate scores among the adaptive OPD baselines, by 1.13/1.56 points. Similar improvements are observed with the 0.6B student, where MetaOPD exceeds OPD by 1.99/5.97 points in Avg@8/Pass@8. The aggregate gains are also broadly reflected at the benchmark level: relative to OPD, MetaOPD improves Pass@8 on all twelve benchmark–scale pairs and Avg@8 on eleven of twelve. The only Avg@8 exception occurs on Minerva Math with the 1.7B student, where MetaOPD is 0.09 points below OPD. On AIME25 with the 1.7B student, MetaOPD obtains the highest Avg@8 while TIP achieves a higher Pass@8, indicating that the two metrics can favor different methods on individual benchmarks.

Out-of-Domain Generalization. We further evaluate the MATH-trained Qwen3-1.7B-Base on GPQA-Diamond, MMLU Pro, and AlpacaEval 2.0, without additional training. Table 2 reports the complete results, with the evaluation protocol provided in Appendix E. MetaOPD achieves the highest scores on both GPQA-Diamond metrics, reaching 27.27 Avg@8 and 78.28 Pass@8, compared with 26.20 and 77.78 for OPD. On MMLU Pro, MetaOPD attains a Pass@1 of 42.30, exceeding OPD by 0.73 percentage points and yielding the highest score among the compared methods. On AlpacaEval 2.0, the improvement over OPD is more limited: MetaOPD obtains 15.98 LC-WR and 18.08 WR, compared with 15.96 and 17.97 for OPD. Overall, these results indicate that the gains from MetaOPD are not confined to the MATH evaluation domain and can transfer to unseen reasoning tasks.

4.3 Ablations and Controls

How reference solutions are used. We further examine whether the gains of MetaOPD can be attributed to direct supervision from the reference solutions used in the meta-objective. We compare against SFT on reference solutions and variants that add the SFT loss directly to the student objective. The reference coefficient is determined by matching the initial gradient norms of the OPD and



(a) Strength sweep

(b) Effect of adding the SFT loss

Figure 3: **Effects of adding SFT to OPD and MetaOPD.** Six-benchmark mean Avg@8/Pass@8 changes (percentage points) for Qwen3-1.7B-Base. (a) Adding the SFT loss to OPD at half, equal, and double the gradient-matched reference coefficient $\gamma^\star \approx 2.16$. (b) Adding the SFT loss at $\gamma^\star$ to OPD or MetaOPD. Changes are relative to each method without the added SFT loss.

SFT losses using training data only, without benchmark-based tuning (Appendix B.6). SFT alone underperforms both OPD and MetaOPD in aggregate performance (Table 9). Moreover, adding the SFT loss to uniform OPD increases Pass@8 but consistently reduces Avg@8 across all three tested coefficients (Figure 3(a)), whereas MetaOPD improves both metrics. Adding the same SFT loss to MetaOPD further decreases Avg@8 and Pass@8 by 5.87 and 6.72 points, respectively (Figure 3(b)). These controls show that the tested forms of direct supervision do not reproduce MetaOPD's gains in both metrics, supporting the use of validation loss after the student update to optimize the token-weighting rule.

Combining token descriptors. We examine the contribution of different token descriptors by restricting the weighting network to individual feature groups while keeping its architecture and the meta-training protocol unchanged. The full descriptor achieves the highest aggregate performance, outperforming the strongest single-group variants by 0.57 points in Avg@8 and 3.20 points in Pass@8 Table 3. Among the restricted variants, teacher entropy yields the highest Avg@8, whereas the teacher–student log-probability gap yields the highest Pass@8. These results indicate that no individual descriptor fully accounts for the performance of MetaOPD and that combining complementary token-level signals provides a more effective basis for learning the weighting rule. Per-benchmark results and the corresponding input masks are provided in Table 8 and Appendix B.6.

Table 3: Feature-ablation Avg. scores (%).

Variant	Avg@8	Pass@8
Uniform OPD	25.07	40.41
Position only	25.21	39.32
Teacher entropy only	<u>26.75</u>	43.25
Log-probability gap only	25.42	<u>43.62</u>
Top-32 profiles only	25.70	42.24
MetaOPD (all features)	**27.32**	**46.82**

5 Conclusion

On-policy distillation provides dense teacher feedback on student rollouts, but local uncertainty or disagreement alone does not determine whether assigning a token a higher weight within the OPD loss will improve student learning. We presented MetaOPD, a one-step bilevel framework that learns to allocate token-level teacher supervision using the student's validation loss after a virtual distillation update. By differentiating this loss on reference solutions through the virtual update, MetaOPD trains a shared weighting network whose token weights guide the student's actual OPD update on its own rollouts. Across six mathematical-reasoning benchmarks and two Qwen3 student scales, MetaOPD achieves the highest six-benchmark averages for Avg@8 and Pass@8 among the compared methods. Ablations and one-step update comparisons support learning token weights within the OPD loss using feedback on the updated student. Extending these findings to broader models and tasks, relating local update quality to long-horizon learning, and reducing meta-update cost remain open directions.

AI USE STATEMENT

OpenAI Codex assisted with research methodology and experimental design, implementation and mathematical checks, manuscript revision and translation, and analysis and presentation of author-supplied results. It did not generate experimental observations or training data. The authors are responsible for verifying the experiments, reviewing all assisted work, and approving the manuscript and artifacts.

ETHICS STATEMENT

This work uses existing models and datasets to study on-policy distillation. Distilled students may inherit biases, errors, or harmful behaviors from their teachers; improved benchmark performance does not establish safety. Downstream use should include appropriate safety evaluation and comply with the licenses and terms of use of the underlying models and datasets.

REPRODUCIBILITY STATEMENT

Section 4 specifies the setup and metrics; Appendix B details configurations and costs, and Appendix A.4 derives the meta-gradient. Appendices C–D give diagnostic protocols, aggregation, and confidence intervals; Appendix E reports out-of-domain evaluation. Code and configurations will accompany the release.

REFERENCES

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *International Conference on Learning Representations*, volume 2024, pp. 21246–21263, 2024.

Mislav Balunović, Jasper Dekoninck, Ivo Petrov, Nikola Jovanović, and Martin Vechev. Matharena: Evaluating llms on uncontaminated math competitions. *Advances in Neural Information Processing Systems*, 38, 2025.

Yann Dubois, Balázs Galambosi, Percy Liang, and Tatsunori B Hashimoto. Length-controlled alpacaeval: A simple way to debias automatic evaluators. *arXiv preprint arXiv:2404.04475*, 2024.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. Minillm: Knowledge distillation of large language models. In *International Conference on Learning Representations*, volume 2024, pp. 32694–32717, 2024.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

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

Nathan Hu, Eric Mitchell, Christopher D Manning, and Chelsea Finn. Meta-learning online adaptation of language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 4418–4432, 2023.

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. *arXiv preprint arXiv:2603.07079*, 2026.

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.

Jongwoo Ko, Sungnyun Kim, Tianyi Chen, and Se-Young Yun. Distillm: Towards streamlined distillation for large language models. *arXiv preprint arXiv:2402.03898*, 2024.

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

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. Solving quantitative reasoning problems with language models. *Advances in neural information processing systems*, 35:3843–3857, 2022.

Mengyang Li, Xudong Zhou, and Pinlong Zhao. Learning temporally-aware sample weights for preference optimization. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 12361–12377, 2026a.

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

Yuying Li, Leqi Zheng, Yongzi Yu, Wenrui Zhou, Xuchang Zhong, Xing Hu, Jing Jin, Hangjie Yuan, and Tao Feng. Filter, then reweight: Rethinking optimization granularity in on-policy distillation. *arXiv preprint arXiv:2606.02684*, 2026c.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, volume 2024, pp. 39578–39601, 2024.

Alexander Lin, Jeremy Wohlwend, Howard Chen, and Tao Lei. Autoregressive knowledge distillation through imitation learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6121–6133, 2020.

Xiaogeng Liu, Xinyan Wang, Yingzi Ma, Yechao Zhang, and Chaowei Xiao. When are teacher tokens reliable? position-weighted on-policy self-distillation for reasoning. *arXiv preprint arXiv:2605.21606*, 2026.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Rui Pan, Dylan Zhang, Hanning Zhang, Xingyuan Pan, Minrui Xu, Jipeng Zhang, Renjie Pi, Xiaoyu Wang, and Tong Zhang. ScaleBiO: Scalable bilevel optimization for LLM data reweighting. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 31959–31982, 2025.

Zikun Qu, Min Zhang, Mingze Kong, Zhiwei Shang, Zhengyu Chen, Yikun Ban, Shuang Qiu, and Zhongxiang Dai. Spot: Sparse probing and outcome calibration for on-policy distillation. *arXiv preprint arXiv:2608.04419*, 2026.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. *arXiv preprint arXiv:2311.12022*, 2023.

Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. In *International conference on machine learning*, pp. 4334–4343. PMLR, 2018.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. Meta-weight-net: Learning an explicit mapping for sample weighting. *Advances in neural information processing systems*, 32, 2019.

Yuanyi Wang, Su Lu, Yanggan Gu, Pengkai Wang, Yifan Yang, Zhaoyi Yan, Congkai Xie, Jianmin Wu, and Hongxia Yang. Not all disagreement is learnable: Token teachability in on-policy distillation. *arXiv preprint arXiv:2605.26844*, 2026.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Advances in Neural Information Processing Systems*, 37: 95266–95290, 2024.

Yuqiao Wen, Zichao Li, Wenyu Du, and Lili Mou. F-divergence minimization for sequence-level knowledge distillation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 10817–10834, 2023.

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

Haolei Xu, Xiaowen Xu, Haiwen Hong, Zixuan Ni, Hongxing Li, Yiwen Qiu, Weiming Lu, and Yongliang Shen. Pass the baton: Trajectory-relayed on-policy distillation. *arXiv preprint arXiv:2607.26057*, 2026a.

Yuanda Xu, Hejian Sang, Zhengze Zhou, Ran He, Zhipeng Wang, and Alborz Geramifard. Tip: Token importance in on-policy distillation. *arXiv preprint arXiv:2604.14084*, 2026b.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

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

Yang Yu, Kai Han, Hang Zhou, Yehui Tang, Kaiqi Huang, Yunhe Wang, and Dacheng Tao. LLM data selection and utilization via dynamic bi-level optimization. *arXiv preprint arXiv:2507.16178*, 2025.

Zichao Yu, Chengzhi Yu, Shengze Xu, Yujin Han, Bingqing Jiang, Xu Wang, and Difan Zou. Mismatch matters: On-policy distillation beyond token agreement. *arXiv preprint arXiv:2608.09836*, 2026.

Rongzhi Zhang, Chenwei Zhang, Xinyang Zhang, Liang Qiu, Haoming Jiang, Yuchen Zhuang, Qingru Zhang, Hyokun Yun, Xian Li, Bing Yin, et al. DORM: Preference data weights optimization for reward modeling in LLM alignment. In *EMNLP (Findings)*, pp. 22721–22739, 2025.

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

Yifan Zhang and Team Math-AI. American invitational mathematics examination (aime). *Mathematical Association of America (MAA)*, 2025.

A Mathematical Details

A.1 Sampled-OPD Gradient and Clipping

We first expand the log-probabilities used in Equation (2). Let μ map student action identities to the teacher view, and let $\mathcal{T}(s)$ convert a visited student prefix to its teacher context. The conversion preserves prompt tokens and follows the completion-side EOS convention in Appendix B.5; stored student actions remain unchanged. Then

$$\begin{aligned} u_{i,t}^{\mathrm{old}} &= \log \pi_{\theta_{\mathrm{old}}}(a_{i,t} \mid s_{i,t}), \\ u_{i,t}^{\theta} &= \log \pi_{\theta}(a_{i,t} \mid s_{i,t}), \\ u_{i,t}^{\mathrm{T}} &= \log \pi_{\mathrm{T}}(\mu(a_{i,t}) \mid \mathcal{T}(s_{i,t}))\,. \end{aligned} \tag{18}$$

Each log-probability is the selected logit minus the $\mathrm{logsumexp}$ over the model's complete vocabulary.

Now fix a visited prefix s, and let $\mathcal{V}_{\mathrm{S}}$ be the student action vocabulary. Because μ need not be bijective, the pulled-back teacher scores do not necessarily sum to one over $\mathcal{V}_{\mathrm{S}}$. Define their normalizer and the induced teacher-score distribution as

$$Z_{\mu}(s) = \sum_{a \in \mathcal{V}_{\mathrm{S}}} \pi_{\mathrm{T}}(\mu(a) \mid \mathcal{T}(s)), \qquad \pi_{\mathrm{T}}^{\mu}(a \mid s) = \frac{\pi_{\mathrm{T}}(\mu(a) \mid \mathcal{T}(s))}{Z_{\mu}(s)}.$$

Abbreviate $u^{\theta}(a) = \log \pi_{\theta}(a \mid s)$ and $u^{\mathrm{T}}(a) = \log \pi_{\mathrm{T}}(\mu(a) \mid \mathcal{T}(s))$. Both $\mathcal{T}(s)$ and $Z_{\mu}(s)$ are fixed with respect to the student parameters in this differentiation. Then $\log \pi_{\mathrm{T}}^{\mu}(a \mid s) = u^{\mathrm{T}}(a) - \log Z_{\mu}(s)$. The corresponding full-vocabulary reverse KL is

$$\mathrm{D}_{\mathrm{KL}}(\pi_{\theta}(\cdot \mid s) \,\|\, \pi_{\mathrm{T}}^{\mu}(\cdot \mid s)) = \mathbb{E}_{a \sim \pi_{\theta}}\left[u^{\theta}(a) - u^{\mathrm{T}}(a) + \log Z_{\mu}(s)\right].$$

Differentiating both the sampling distribution and its log-density gives

$$\begin{aligned} \nabla_{\theta} \mathrm{D}_{\mathrm{KL}}(\pi_{\theta} \| \pi_{\mathrm{T}}^{\mu}) &= \mathbb{E}_{a \sim \pi_{\theta}}\left[\left(u^{\theta}(a) - u^{\mathrm{T}}(a) + \log Z_{\mu}(s) + 1\right) \nabla_{\theta} \log \pi_{\theta}(a \mid s)\right] \\ &= \mathbb{E}_{a \sim \pi_{\theta}}\left[\left(u^{\theta}(a) - u^{\mathrm{T}}(a)\right) \nabla_{\theta} \log \pi_{\theta}(a \mid s)\right], \end{aligned} \tag{19}$$

because both 1 and $\log Z_{\mu}(s)$ are action-independent at the fixed prefix and the expected score $\mathbb{E}_{\pi_{\theta}}[\nabla_{\theta} \log \pi_{\theta}(a \mid s)]$ is zero. At $\theta = \theta_{\mathrm{old}}$, $r = 1$ and

$$\nabla_{\theta}[-A(a) r(a;\theta)]|_{\theta=\theta_{\mathrm{old}}} = \left(u^{\mathrm{old}}(a) - u^{\mathrm{T}}(a)\right) \nabla_{\theta} \log \pi_{\theta}(a \mid s)|_{\theta=\theta_{\mathrm{old}}}\,,$$

so a behavior-policy action provides an unbiased single-sample estimator of Equation (19). This statement is conditional on the visited prefix and does not differentiate the trajectory distribution.

Away from the rollout policy, the objective with the cached A held fixed has expectation

$$\mathbb{E}_{a \sim \pi_{\theta_{\mathrm{old}}}}[-A(a) r(a;\theta)] = \mathbb{E}_{a \sim \pi_{\theta}}\left[u^{\mathrm{old}}(a) - u^{\mathrm{T}}(a)\right].$$

It therefore differs from the current reverse KL, whose score coefficient contains $u^{\theta}(a) - u^{\mathrm{T}}(a)$. The importance ratio supports a local update from cached behavior-policy actions; it does not restore exact reverse-KL equivalence at an arbitrary θ.

The active branch of the clipped loss depends on the sign of A. For $A > 0$, it is $-A \min(r, 1 + \epsilon_h)$, so an excessive probability increase is clipped. For $A < 0$, it is $-A \max(r, 1 - \epsilon_l)$, so an excessive probability decrease is clipped. At a clipping boundary, automatic differentiation supplies the implementation-defined subgradient.

A.2 Score Centering and Response Normalization

For response i, let $n_i = \sum_t m_{i,t}$. Under the convention following Equation (4), all normalized weights are zero when $n_i = 0$. Below, assume $n_i > 0$ and define

$$\overline{w}_i = \frac{1}{n_i} \sum_t m_{i,t} \widetilde{w}_{i,t}.$$

Then Equation (4) is $w_{i,t} = \widetilde{w}_{i,t}/\overline{w}_i$ on valid positions, which immediately gives

$$\frac{1}{n_i}\sum_t m_{i,t} w_{i,t} = 1.$$

With $\widetilde{w} \in (1-\rho, 1+\rho)$, the normalized weights satisfy

$$w_{i,t} \in \left(\frac{1-\rho}{1+\rho}, \frac{1+\rho}{1-\rho}\right).$$

With $\rho = 0.25$, the normalized range is $w_{i,t} \in (0.6, 5/3)$, even though the final weights need not remain inside the raw interval $(0.75, 1.25)$.

For valid positions t and j,

$$\frac{\partial w_{i,t}}{\partial \widetilde{w}_{i,j}} = \frac{\mathbb{I}[t=j]}{\overline{w}_i} - \frac{\widetilde{w}_{i,t}}{n_i \overline{w}_i^2},$$

which makes the within-response coupling explicit. Combining this derivative with Equation (15) gives the gradient with respect to the unnormalized weights in Equation (16), expanded here for a valid raw weight:

$$\frac{\partial \mathcal{L}_{\text{out}}}{\partial \widetilde{w}_{i,j}} = -\frac{1}{N_B \overline{w}_i}\left(c_{i,j} - \frac{1}{n_i}\sum_t m_{i,t} w_{i,t} c_{i,t}\right).$$

Here $c_{i,t}$ is defined in Equation (15); its learning-rate dependence is already included in $\boldsymbol{d}$. Thus normalization subtracts the response-level weighted mean from $c_{i,j}$, rather than using its sign alone. Invalid weights are zero, so a completely masked response contributes to neither the weighted loss nor the weight regularizer. The implementation also clamps valid counts below by one and rejects optimizer minibatches with $N_B = 0$.

Score centering introduces a second coupling before the residual map. On the valid positions of response i, let $\boldsymbol{z}_i \in \mathbb{R}^{n_i}$ collect the network scores and define

$$P_i = I - \frac{1}{n_i}\mathbf{1}\mathbf{1}^\top, \qquad \boldsymbol{z}_i^\circ = P_i \boldsymbol{z}_i, \qquad D_i = \operatorname{diag}\big(\rho[1 - \tanh^2(z_{i,t}^\circ)]\big).$$

Writing J_i for the normalization Jacobian above, the complete derivative is

$$\frac{\partial \boldsymbol{w}_i}{\partial \boldsymbol{z}_i} = J_i D_i P_i, \qquad \nabla_{\boldsymbol{z}_i}\mathcal{L}_{\text{out}} = -\frac{1}{N_B} P_i D_i J_i^\top \boldsymbol{c}_i. \tag{20}$$

Both P_i and D_i are symmetric. In particular, the score derivative includes the mean-score derivative; detaching that mean would change the meta-gradient. A common score shift is annihilated by P_i, and a response with one valid token has constant weight one. The same Jacobian propagates the regularizer derivative, with $\nabla_{\boldsymbol{w}_i}\mathcal{R}_w = 2(\boldsymbol{w}_i - \mathbf{1})/N_B$. The implementation safeguards the normalization denominator by 10^{-8}; since $\sum_t m_{i,t}\widetilde{w}_{i,t} > n_i(1-\rho)$, this safeguard is inactive for nonempty responses.

A.3 AdamW Virtual Update

Consider one optimizer minibatch and suppress its training index. For each trainable parameter tensor θ_j, the current real optimizer supplies first and second moments $\boldsymbol{m}_j, \boldsymbol{v}_j$, a completed-step count τ_j, and its parameter-group settings $(\alpha_j, \beta_{1,j}, \beta_{2,j}, \varepsilon_j, \lambda_j)$: learning rate, moment decays, denominator epsilon, and decoupled weight decay. These quantities form S in Equation (6). The virtual step reads them as constants with respect to η. Zero initialization is used only if the corresponding real optimizer state is uninitialized; moments and step counts are not reset at subsequent virtual steps. We assume the standard settings $0 \le \beta_{1,j} < 1$, $0 < \beta_{2,j} < 1$, and $\varepsilon_j > 0$.

Global gradient clipping. The input to clipping is the full-model gradient of the global token-mean inner loss, after microbatch accumulation and distributed reduction. Let $G_{\max} > 0$ be the real update's norm threshold and $\varepsilon_{\text{clip}} \ge 0$ its numerical safeguard. For $n = \|\boldsymbol{g}_{\text{in}}(\eta)\|_2 > 0$, define

$$\kappa_{\text{clip}} = \min\left(1, \frac{G_{\max}}{n + \varepsilon_{\text{clip}}}\right), \qquad \overline{\boldsymbol{g}}_{\text{in}}(\eta) = \mathcal{C}(\boldsymbol{g}_{\text{in}}(\eta)) = \kappa_{\text{clip}} \boldsymbol{g}_{\text{in}}(\eta). \tag{21}$$

The zero gradient maps to zero. Clipping uses one norm over all trainable student parameters, including parameters in different optimizer groups; it is not applied separately to each microbatch or parameter tensor. On a smooth branch, its Jacobian is

$$J_{\mathcal{C}} = \begin{cases} I, & n + \varepsilon_{\text{clip}} < G_{\max}, \\ \kappa_{\text{clip}} I - \dfrac{G_{\max}}{n(n+\varepsilon_{\text{clip}})^2} \boldsymbol{g}_{\text{in}}(\eta)\boldsymbol{g}_{\text{in}}(\eta)^\top, & n + \varepsilon_{\text{clip}} > G_{\max},\ n > 0. \end{cases} \tag{22}$$

At the switching boundary, the derivative follows the branch convention of the real clipping operation. The rank-one term accounts for the dependence of the clipping coefficient on the weights through $\boldsymbol{g}_{\text{in}}(\eta)$. Detaching that coefficient would omit this term. At the origin, $J_{\mathcal{C}} = \min(1, G_{\max}/\varepsilon_{\text{clip}})I$ for $\varepsilon_{\text{clip}} > 0$, and $J_{\mathcal{C}} = I$ when the safeguard is zero. If clipping is disabled, $\mathcal{C}$ and $J_{\mathcal{C}}$ are the identity.

Moment update and parameter map. For standard AdamW, the virtual moments, bias corrections, and virtual student parameters are

$$\begin{aligned} \boldsymbol{m}_j^+ &= \beta_{1,j}\boldsymbol{m}_j + (1-\beta_{1,j})\overline{\boldsymbol{g}}_{\text{in},j}(\eta), \\ \boldsymbol{v}_j^+ &= \beta_{2,j}\boldsymbol{v}_j + (1-\beta_{2,j})\overline{\boldsymbol{g}}_{\text{in},j}(\eta)^{\,2}, \\ \widehat{\boldsymbol{m}}_j &= \frac{\boldsymbol{m}_j^+}{1-\beta_{1,j}^{\tau_j+1}}, \qquad \widehat{\boldsymbol{v}}_j = \frac{\boldsymbol{v}_j^+}{1-\beta_{2,j}^{\tau_j+1}}, \\ \theta_j' &= (1-\alpha_j\lambda_j)\theta_j - \alpha_j \frac{\widehat{\boldsymbol{m}}_j}{\sqrt{\widehat{\boldsymbol{v}}_j} + \varepsilon_j}. \end{aligned} \tag{23}$$

All products, squares, square roots, and divisions within a parameter tensor are elementwise. The learning rate α_j is read from the current real parameter group, including scheduled decay. Bias correction uses that tensor's next step $\tau_j + 1$. The expressions above create temporary values; they do not overwrite θ_j, $\boldsymbol{m}_j$, $\boldsymbol{v}_j$, or τ_j. Only the later real student step advances the stored moments and step counts.

For clarity, the AdamW Jacobian can also be written coordinatewise. Let

$$\zeta_{1,j} = \frac{1-\beta_{1,j}}{1-\beta_{1,j}^{\tau_j+1}}, \qquad \zeta_{2,j} = \frac{1-\beta_{2,j}}{1-\beta_{2,j}^{\tau_j+1}}, \qquad \sigma_{j,k} = \sqrt{\widehat{v}_{j,k}},$$

where k indexes a coordinate of tensor j. For $\sigma_{j,k} > 0$,

$$\frac{\partial \theta_{j,k}'}{\partial \overline{g}_{\text{in},j,k}(\eta)} = -\alpha_j \left[\frac{\zeta_{1,j}}{\sigma_{j,k} + \varepsilon_j} - \frac{\zeta_{2,j}\widehat{m}_{j,k}\overline{g}_{\text{in},j,k}(\eta)}{\sigma_{j,k}(\sigma_{j,k} + \varepsilon_j)^2} \right]. \tag{24}$$

The off-diagonal entries of $J_{\mathcal{A}}$ are zero. The second term differentiates the new second moment; treating AdamW as a fixed preconditioner would discard it. At a zero-gradient coordinate with zero first and second moments, the composite map has derivative $-\alpha_j\zeta_{1,j}/\varepsilon_j$, obtained by continuity rather than division by $\sigma_{j,k} = 0$. Decoupled weight decay has no direct derivative with respect to the gradient input because θ_j is fixed. It still changes θ', and hence the outer gradient, so it must remain in the forward virtual map.

A.4 One-Step Hypergradient and VJP–JVP Form

We derive the token decomposition in §3.3 from the complete meta-gradient in Equation (10). Vectorize the student and weighting parameters as $\theta \in \mathbb{R}^p$ and $\eta \in \mathbb{R}^q$. Gradients are column vectors; the Jacobian of a map from $\mathbb{R}^q$ to $\mathbb{R}^p$ has shape $p \times q$, with rows indexing outputs and columns indexing inputs. We use the fixed-input convention of §3.2: current student parameters, real optimizer state, rollout actions, masks, old/teacher caches, and detached descriptors are held fixed.

Only the normalized weights in Equation (13) depend on η. Since $\boldsymbol{g}_{i,t} \in \mathbb{R}^p$ and $\nabla_\eta w_{\eta,i,t} \in \mathbb{R}^q$, differentiating each weighted term gives the outer product

$$G_\eta := \frac{\partial \boldsymbol{g}_{\text{in}}(\eta)}{\partial \eta} = \frac{1}{N_B}\sum_{i,t} m_{i,t}\boldsymbol{g}_{i,t}\left(\nabla_\eta w_{\eta,i,t}\right)^\top \in \mathbb{R}^{p\times q}, \qquad \frac{\partial \theta'}{\partial \eta} = J_{\mathcal{A}}J_{\mathcal{C}}G_\eta. \tag{25}$$

The two optimizer Jacobians are $p \times p$, and $\boldsymbol{g}_{\text{out}} \in \mathbb{R}^p$. Thus the validation-loss term of Equation (10) is the q-vector

$$\nabla_\eta \mathcal{L}_{\text{out}} = \left(\frac{\partial \boldsymbol{g}_{\text{in}}(\eta)}{\partial \eta} \right)^\top J_{\mathcal{C}}^\top J_{\mathcal{A}}^\top \boldsymbol{g}_{\text{out}} = G_\eta^\top J_{\mathcal{C}}^\top J_{\mathcal{A}}^\top \boldsymbol{g}_{\text{out}}. \tag{26}$$

Using $J_{\mathcal{C}}^\top J_{\mathcal{A}}^\top \boldsymbol{g}_{\text{out}} = -\boldsymbol{d}$ from Equation (14) and expanding $G_\eta^\top$ yields

$$\nabla_\eta \mathcal{L}_{\text{out}} = -\frac{1}{N_B} \sum_{i,t} m_{i,t} \left(\nabla_\eta w_{\eta,i,t} \right) \boldsymbol{g}_{i,t}^\top \boldsymbol{d} = -\frac{1}{N_B} \sum_{i,t} m_{i,t} c_{i,t} \nabla_\eta w_{\eta,i,t}. \tag{27}$$

Here $\boldsymbol{d} \in \mathbb{R}^p$ and $c_{i,t} = \boldsymbol{g}_{i,t}^\top \boldsymbol{d}$ is scalar, establishing Equation (15).

To verify the gradient of Equation (17), fix an evaluation point η_0 and write $c_{i,t}^0 = c_{i,t}(\eta_0)$. The VJP–JVP procedure in §3.3 supplies these values. Differentiating the objective while holding them fixed gives

$$\begin{aligned} \nabla_\eta \widehat{\mathcal{L}}_{\text{meta}}^{(r)}(\eta; B, \boldsymbol{c}^0)\Big|_{\eta_0} &= -\frac{1}{N_B} \sum_{i,t} m_{i,t} c_{i,t}^0 \, \nabla_\eta w_{\eta,i,t}\big|_{\eta_0} + \lambda_w^{(r)} \, \nabla_\eta \mathcal{R}_w\big|_{\eta_0} \\ &= \nabla_\eta \mathcal{J}_{\text{meta}}\big|_{\eta_0} . \end{aligned}$$

The second equality uses Equation (27). Exactness concerns the specified single-step map, including its new moments and clipping coefficient, under the fixed-input convention above. At clipping boundaries, the derivative follows the corresponding automatic-differentiation convention.

B Implementation and Reproducibility Details

The inner rollout stream and the reference-solution stream are separately shuffled iterators over the MATH training split. They are not constrained to be sample-disjoint, and the experimental setup does not create a separate validation pool for meta-updates. Evaluation examples do not update either the student parameters θ or the weighting network parameters η.

B.1 Token Descriptor

Table 4 expands the 74-dimensional descriptor defined in Section 3.1, following its detachment and sampled-loss conventions. Top-k_f token identities are used only to compute the top-1 match and top-k_f overlap; the identities themselves are not network inputs.

Table 4: Composition of the 74-dimensional weighting network descriptor, with $k_f = 32$.

Feature group	Features	Dim.	Detached
Sampled action	Current-student log-probability, teacher log-probability, and teacher-minus-current log-probability gap	3	Yes
Uncertainty	Vocabulary-normalized student and teacher entropy	2	Yes
Top-1 behavior	Student and teacher top-1 probability; student–teacher top-1 identity match	3	Yes
Position and overlap	Position normalized by the common completion-window width; student–teacher top-32 set overlap	2	Yes
Ranked probabilities	Student top-32 and teacher top-32 full-vocabulary-normalized log-probability values	64	Yes
Total	Ten scalar statistics plus two ranked top-32 vectors	74	Yes

For a common completion tensor of width L_{win}, the position feature is zero when $L_{\text{win}} = 1$ and otherwise equals $(t-1)/(L_{\text{win}}-1)$, using the one-based token index in Equation (1). The same position grid is shared across rows before masking; it is not renormalized by each response's number of valid completion tokens.

Let p^{S} and p^{T} be the full-vocabulary student and teacher distributions at their respective context views. The entropy features are

$$h^{\mathrm{S}} = \frac{-\sum_{a\in\mathcal{V}_{\mathrm{S}}} p^{\mathrm{S}}(a)\log p^{\mathrm{S}}(a)}{\log|\mathcal{V}_{\mathrm{S}}|}, \qquad h^{\mathrm{T}} = \frac{-\sum_{a\in\mathcal{V}_{\mathrm{T}}} p^{\mathrm{T}}(a)\log p^{\mathrm{T}}(a)}{\log|\mathcal{V}_{\mathrm{T}}|}.$$

Let S_{32} and T_{32} contain the top-32 token identities in the two distributions. With teacher identities canonicalized as in Appendix B.5, the overlap feature is

$$\mathrm{Overlap}_{32} = \frac{1}{32}\sum_{a\in S_{32}} \mathbb{I}[a \in \mathrm{Canon}(T_{32})]\,.$$

The denominator remains 32 even if canonicalization merges teacher identities; this is a student-set coverage fraction, not Jaccard overlap.

The weighting network has linear widths 74–256–64–1, SiLU activations after the first two linear layers, and dropout probability zero. Its final scalar head is zero-initialized. Consequently, the initial raw and response-normalized weights are exactly one on all valid tokens. The raw weights use $\rho = 0.25$, and the normalized-weight regularizer starts at $\lambda_w^{(0)} = 10^{-3}$ and follows Appendix B.4.

B.2 Training and Evaluation Configuration

Table 5 summarizes the shared training and evaluation configuration for the main and ablation experiments. Settings for the weighting network and virtual updates apply only to variants with meta-updates. Settings for reference-solution data apply to variants that use reference solutions for supervision. All learned-weight variants use response-centered scores and the schedules below, except for the factor explicitly changed in an ablation. The ablations use the same optimizer settings as the main experiments; virtual student steps follow the actual student's AdamW update rule.

Baseline settings. KD distills the student on teacher-generated sequences. GRPO uses standard answer-correctness rewards with a group size of eight. EOPD uses the method-specific settings reported in its paper. Appendix B.2.1 details how the FiRe-OPD, REOPOLD, and TIP allocation rules are implemented within the shared sampled-OPD framework.

B.2.1 Token-Allocation Baseline Implementations

To study supervision allocation under a common OPD training framework, we implement the token-selection and weighting rules of FiRe-OPD (Li et al., 2026c), REOPOLD (Ko et al., 2026), and TIP (Xu et al., 2026b) on top of the sampled-OPD surrogate in Equation (3). The underlying surrogate form, student optimizer, and training schedule are shared; method-specific selection, weighting, reward processing, and loss reduction are specified below. This design focuses the comparison on supervision allocation while controlling for differences in the original methods' underlying OPD objectives.

Shared implementation. Selection statistics use the frozen rollout-time student and teacher predictions and are detached from the student update. Batch statistics and rankings are computed over valid completion tokens in the full 32-response optimizer minibatch, across all microbatches and student ranks; response-level selection is identified below. The models, data, rollout batch size, ratio clipping, and decoding follow Table 5. All three baselines use AdamW with zero weight decay, cosine decay advanced once per rollout, BF16 forward computation, and FP32 student parameters and sampled-action log-probabilities. They use no learned weighting network, meta-update, or reference-solution loss. Training runs for 696 student updates, and evaluation uses the final checkpoint at update 696.

FiRe-OPD. Responses are ranked by mean teacher log-probability of their sampled tokens. We remove the lowest-scoring $\lfloor 0.2|B|\rfloor$ responses (six for $|B| = 32$). Let h^{T} and h^{S} denote vocabulary-normalized teacher and rollout-student entropies. Before filtering, their respective maxima are computed over all valid tokens in B. Teacher confidence and student confusion are

$$c^{\mathrm{T}}_{i,t} = 1 - \frac{h^{\mathrm{T}}_{i,t}}{\max_B h^{\mathrm{T}}}, \qquad c^{\mathrm{S}}_{i,t} = \frac{h^{\mathrm{S}}_{i,t}}{\max_B h^{\mathrm{S}}}.$$

Retained-token weights are proportional to $(1 + c^{\mathrm{T}}_{i,t})(1 + c^{\mathrm{S}}_{i,t})$, normalized to mean one within each retained response. Entropy denominators are lower-bounded by FP32 machine epsilon; removed responses contribute neither loss nor denominator tokens.

Table 5: Shared training and evaluation configuration, including MetaOPD-specific settings.

Setting	Value
Student / teacher models	Qwen3-0.6B-Base or Qwen3-1.7B-Base / frozen Qwen3-8B
Training data / MetaOPD reference-solution data	MATH train / MATH train
MetaOPD rollout / reference sampling	Separate shuffled streams; overlap allowed; no meta split
Shared epochs / training seed	3 / 42
Shared rollout and update schedule	174 rollout iterations; four student updates per rollout; 696 student updates
Shared batch sizes	128 rollout / 32 student optimizer; global reference batch 32 where used
Shared student optimizer / learning rate	AdamW / 3×10^{-6}, cosine decay
MetaOPD meta optimizer / initial learning rate	AdamW / 2.5×10^{-5}; scaled by the student learning-rate ratio
MetaOPD virtual optimizer / state	Same AdamW rule as the actual student update; current student optimizer state, without committing changes
MetaOPD virtual learning rates	Current real student optimizer rates, including scheduled decay
Student gradient clipping	Full-model global norm, threshold 1.0 in virtual and actual steps
Sampled-OPD ratio clipping	$(\epsilon_l, \epsilon_h) = (0.2, 0.2)$
MetaOPD residual / initial regularizer	$\rho = 0.25$ / $\lambda_w^{(0)} = 10^{-3}$; regularizer follows the student learning-rate ratio
MetaOPD score centering / meta updates	Valid-response mean subtracted before $\tanh$ / one per student update
Shared training decoding	Temperature 1.0; top-p 1.0; maximum completion 4096
Math evaluation decoding	Eight samples; temperature 1.0; top-p 0.8; maximum response 8192
Allocation for reported GPU-hours	Eight NVIDIA A100 80GB GPUs: four student DDP and four paired teacher GPUs

Table 6: Method-specific allocation settings within the shared sampled-OPD framework. Selection scopes and loss normalization are specified in the text.

Method	Configuration
FiRe-OPD	Trajectory drop fraction 0.2; teacher-confidence and student-confusion coefficients both 1.0
REOPOLD	Reward parameter $\lambda = 0.3$; phase switch at update 232; phase-II entropy keep fraction 0.2
TIP	Soft-OR; response-level keep fraction 0.5; entropy clipping at the 98th percentile; exact-KL computation in 64-token chunks

REOPOLD. For the detached reward $R_{i,t} = u_{i,t}^{\mathrm{T}} - u_{i,t}^{\mathrm{old}}$, define $R_{\min} = \log(\lambda)/(1-\lambda)$. We replace $A_{i,t}$ in Equation (3) with $\max(R_{i,t}, R_{\min})$. At zero-based updates 0–231, tokens with $R_{i,t} \geq R_{\min}$ are retained. At updates 232–695, selection instead retains the $\lceil 0.2 N_B \rceil$ highest-student-entropy tokens across the optimizer minibatch. If a response would be empty, its highest-reward token in phase I or highest-entropy token in phase II is retained. This safeguard can raise the actual retained fraction above 0.2 in phase II.

TIP Soft-OR. The two signals are $h_{i,t} = H(\pi_{\theta_{\mathrm{old}}}(\cdot \mid s_{i,t}))/\log|\mathcal{V}|$ and the exact full-vocabulary $d_{i,t} = \mathrm{D}_{\mathrm{KL}}(\pi_{\theta_{\mathrm{old}}} \| \pi_{\mathrm{T}})$ at the corresponding student and teacher context views. The latter is computed from full logits, rather than from the sampled-token log-probability gap. We cap h at its minibatch 98th percentile, then separately min–max normalize the capped entropy and d to obtain $\hat{h}$ and $\hat{d}$. A signal whose range is at most FP32 machine epsilon maps to zero. The score is $s_{i,t} = \hat{h}_{i,t} + \hat{d}_{i,t} - \hat{h}_{i,t}\hat{d}_{i,t}$. Each response retains its $\max(1, \lfloor 0.5 n_i \rfloor)$ highest-scoring valid tokens. Ranking ties in all three methods preserve the gathered input order. Full-vocabulary reverse KL is used for token selection; the student update on retained tokens uses the shared sampled-OPD surrogate in Equation (3) in place of TIP's original full-vocabulary reverse-KL training loss.

Reduction after selection. For all three baselines, the loss is the weighted sum of retained-token surrogate losses divided by the total number of retained tokens in the global optimizer minibatch. FiRe-OPD uses its response-normalized soft weights; REOPOLD and TIP use unit weights on retained tokens and zero weights elsewhere. The binary masks are not rescaled to have mean one over the original response. Thus, filtering changes both the numerator and the token-count denominator.

B.3 Global Token-Mean Reduction

Both the virtual and actual student updates evaluate the weighted OPD loss using the global valid-token count N_B from the complete 32-example optimizer minibatch. The numerator is accumulated across all local microbatches and distributed ranks before division by N_B; the implementation does not average independently normalized microbatch losses. Response-level weight normalization in Equation (4) is applied before this global reduction.

To compute validation loss, we similarly sum token losses on reference solutions over the microbatches and participating student ranks, then divide by the global valid reference-solution token count N_{ref}. Because completion lengths differ, N_B and N_{ref} are generally different even though both global example batches contain 32 examples. The inner gradient is globally reduced before norm clipping and the virtual optimizer map; the outer gradient is globally reduced before the optimizer VJPs. Clipping each rank's partial gradient would define a different virtual update.

B.4 Virtual State and Learning-Rate Convention

As specified in §3.2, training-time virtual steps use the current student AdamW state and scheduled learning rates without committing optimizer or parameter changes.

The motivating comparison in Appendix C starts from zero student moments and step zero. The checkpoint probes in Appendix D restore the trained student optimizer state. Both use a fixed reference learning rate of 3×10^{-6}, whereas training-time virtual steps read the scheduled rate.

Coupled learning-rate schedules. Let $\alpha_g^{(r)}$ be the live student learning rate in trainable parameter group g at global student step r, and let $\alpha_0 = 3 \times 10^{-6}$ be the configured initial rate. The schedule scale and effective meta settings are

$$\chi_r = \min\left(1, \max\left(0, \max_g \frac{\alpha_g^{(r)}}{\alpha_0}\right)\right), \quad \alpha_\eta^{(r)} = \alpha_{\eta,0}\chi_r, \quad \lambda_w^{(r)} = \lambda_w^{(0)}\chi_r, \tag{28}$$

where $\alpha_{\eta,0} = 2.5 \times 10^{-5}$ and $\lambda_w^{(0)} = 10^{-3}$. The student uses cosine decay, advanced once per rollout round; all four optimizer minibatches in that round read its current rate. The weighting network uses AdamW with zero weight decay and is updated at each student step throughout all 696 updates. The schedule values are constants for the current one-step meta-gradient. Neither the virtual step nor the meta-update advances the student scheduler.

B.5 End-of-Sequence Mapping

The action map μ in Equation (18) is the identity except for $\mu(151643) = 151645$, mapping the student EOS identity to the teacher EOS identity. The context conversion $\mathcal{T}$ leaves prompt tokens unchanged and applies this mapping only to the first valid completion occurrence of 151643 in the teacher view. Padding, later invalid positions, and stored student actions are unchanged. The valid sampled EOS action is scored with its teacher identity. Teacher top-k identities are canonicalized back to the student identity before match and overlap features are computed. This convention aligns EOS across model views without introducing a dual-EOS student stop set.

Reference-solution targets use the student EOS token 151643, which is also the sole generation stop token. The teacher-view mapping does not change this student target identity. The SFT, OPD + SFT, and MetaOPD + SFT controls use this same EOS convention.

B.6 Ablation and Resource Details

Variant definitions and SFT loss calibration. Let $\mathcal{L}_{\mathrm{OPD}}^{w}$ denote the OPD loss on student-generated responses with token weights w. In OPD + SFT, every valid token has weight one. In MetaOPD +

Table 7: Ablation-variant mapping. "Outer" means that reference solutions update only the weighting network through the meta objective; "direct" means that their supervised loss enters the student objective.

Variant	Token weighting	Reference use	Student objective
Uniform OPD	Uniform	None	$\mathcal{L}_{\mathrm{OPD}}$
Position only	Position	Outer	$\mathcal{L}^{w}_{\mathrm{OPD}}$
Teacher entropy	Teacher entropy	Outer	$\mathcal{L}^{w}_{\mathrm{OPD}}$
Log-probability gap	$\delta_{i,t}$	Outer	$\mathcal{L}^{w}_{\mathrm{OPD}}$
Top-32 profiles	Student + teacher profiles	Outer	$\mathcal{L}^{w}_{\mathrm{OPD}}$
MetaOPD	All 74 descriptors	Outer	$\mathcal{L}^{w}_{\mathrm{OPD}}$
SFT	None	Direct only	$\mathcal{L}_{\mathrm{SFT}}$
OPD + SFT (half)	Uniform	Direct only	$\mathcal{L}_{\mathrm{OPD}} + 0.5\gamma^{\star}\mathcal{L}_{\mathrm{SFT}}$
OPD + SFT	Uniform	Direct only	$\mathcal{L}_{\mathrm{OPD}} + \gamma^{\star}\mathcal{L}_{\mathrm{SFT}}$
OPD + SFT (double)	Uniform	Direct only	$\mathcal{L}_{\mathrm{OPD}} + 2\gamma^{\star}\mathcal{L}_{\mathrm{SFT}}$
MetaOPD + SFT	All 74 descriptors	Outer + direct	$\mathcal{L}^{w}_{\mathrm{OPD}} + \gamma^{\star}\mathcal{L}_{\mathrm{SFT}}$

SFT, the weighting network assigns the weights, which have mean one over the valid tokens in each response. Let $\mathcal{L}_{\mathrm{SFT}}$ denote the supervised fine-tuning (SFT) loss: the mean token prediction loss on reference solutions under teacher forcing. It has the form of Equation (7), evaluated at the current student parameters rather than the virtual parameters. Each loss is averaged over the valid tokens in its own global batch. Both variants update the student with

$$\mathcal{L}_{\mathrm{student}} = \mathcal{L}^{w}_{\mathrm{OPD}} + \gamma\mathcal{L}_{\mathrm{SFT}}.$$

The coefficient is calibrated once at the common Qwen3-1.7B-Base initialization using 16 disjoint pairs of 32-example OPD and reference-solution batches:

$$\gamma^{\star} = \mathrm{median}_{k=1}^{16} \frac{\|\nabla_{\theta}\mathcal{L}^{(k)}_{\mathrm{OPD}}\|_2}{\|\nabla_{\theta}\mathcal{L}^{(k)}_{\mathrm{SFT}}\|_2 + 10^{-12}}.$$

Thus $\gamma^{\star}$ approximately matches the two gradient scales at initialization; it is applied only to the SFT loss in the student objective, not to the MetaOPD outer objective.

The resulting coefficient is $\gamma^{\star} \approx 2.16$; the sweep uses half, equal, and double this value. Calibration uses training data only and is not selected by benchmark performance. Matching gradient norms does not imply that the two gradients have the same direction or induce equal optimizer updates.

For MetaOPD + SFT, the virtual inner objective remains weighted sampled OPD. The same global reference batch used for the outer objective supplies the SFT loss in the subsequent real student update; this loss is evaluated at the real student parameters.

Feature-ablation controls and results. Table 8 compares uniform OPD, four restricted-input weighting networks, and MetaOPD with all descriptors. The Uniform OPD and MetaOPD rows reuse the Qwen3-1.7B-Base results from Table 1.

The restricted variants retain, respectively, normalized position, vocabulary-normalized teacher entropy, the teacher-minus-student log-probability gap $\delta_{i,t}$, or the two ranked top-32 log-probability profiles. All other descriptor coordinates are zeroed while the 74-dimensional input and 74–256–64–1 architecture remain fixed. All learned variants have 35,713 parameters and follow the shared training protocol with seed 42, the shared virtual-update rule, and 696 student updates. This design compares named input subsets with the full descriptor rather than removing one feature at a time.

Direct-supervision ablations. Table 9 reports SFT, the SFT loss-weight sweep under uniform OPD, and the interaction control that adds the SFT loss to MetaOPD. The Uniform OPD and MetaOPD rows reuse the 1.7B main-table results. All controls use the shared EOS convention, training seed 42, and 696 student updates. The MetaOPD rows retain the shared virtual-update rule; SFT and the uniform-OPD sweep have no meta-update.

Resource accounting. Table 10 compares OPD and MetaOPD with the 1.7B student, training seed 42, and the same eight-GPU allocation. Both runs use 174 rollout iterations, rollout batches of 128,

Table 8: Feature-composition ablation on Qwen3-1.7B-Base (%).

Variant	Avg@8							Pass@8						
	MATH500	AMC23	Minerva	HMMT	AIME24	AIME25	Avg.	MATH500	AMC23	Minerva	HMMT	AIME24	AIME25	Avg.
Uniform OPD	67.23	38.75	27.80	1.67	8.33	6.67	25.07	84.20	67.50	47.43	6.67	20.00	16.67	40.41
Position only	68.50	39.69	26.01	1.67	9.58	5.83	25.21	88.00	67.50	43.75	3.33	20.00	13.33	39.32
Teacher entropy only	68.88	**43.75**	**27.85**	**2.50**	11.25	6.25	26.75	86.60	72.50	47.06	**10.00**	26.67	16.67	43.25
Log-probability gap only	68.50	39.38	26.75	1.67	8.75	7.50	25.42	87.80	75.00	45.59	6.67	26.67	20.00	43.62
Top-32 profiles only	69.05	40.94	27.57	1.67	8.33	6.67	25.70	**88.40**	75.00	46.69	3.33	20.00	20.00	42.24
MetaOPD (all features)	**69.95**	**43.75**	27.71	2.08	**12.08**	**8.33**	**27.32**	88.20	**77.50**	**48.53**	**10.00**	**33.33**	**23.33**	**46.82**

All runs use the same six-benchmark evaluation protocol with eight samples per prompt. Avg. denotes the unweighted average. Bold and underline mark the highest and second-highest distinct scores in each column, including ties. Restricted variants retain only the named descriptor inputs; top-32 profiles concatenate the student's and teacher's ranked log-probability profiles.

Table 9: SFT and combined OPD–SFT controls on Qwen3-1.7B-Base (%).

Variant	Avg@8							Pass@8						
	MATH500	AMC23	Minerva	HMMT	AIME24	AIME25	Avg.	MATH500	AMC23	Minerva	HMMT	AIME24	AIME25	Avg.
Uniform OPD	67.23	38.75	**27.80**	1.67	8.33	6.67	25.07	84.20	67.50	47.43	6.67	20.00	16.67	40.41
SFT	32.85	14.06	11.76	1.25	0.83	0.00	10.13	65.40	52.50	34.93	6.67	6.67	0.00	27.69
OPD + SFT ($0.5\gamma^\star$)	67.45	38.44	23.99	**2.50**	7.92	6.67	24.49	86.60	75.00	43.75	**10.00**	16.67	16.67	41.45
OPD + SFT ($\gamma^\star$)	63.00	33.75	23.53	2.08	7.50	6.25	22.69	86.20	**77.50**	45.59	6.67	23.33	13.33	42.10
OPD + SFT ($2\gamma^\star$)	57.25	27.81	22.75	1.25	8.33	5.42	20.47	86.20	67.50	47.79	**10.00**	20.00	20.00	41.92
MetaOPD	**69.95**	**43.75**	27.71	2.08	**12.08**	**8.33**	**27.32**	**88.20**	**77.50**	**48.53**	**10.00**	**33.33**	**23.33**	**46.82**
MetaOPD + SFT	60.20	28.44	23.81	2.08	7.92	6.25	21.45	86.80	57.50	46.32	6.67	20.00	**23.33**	40.10

All runs use the same six-benchmark evaluation protocol with eight samples per prompt. Avg. denotes the unweighted average; benchmark and Avg. scores are rounded separately. Bold and underline mark the highest and second-highest distinct scores in each column, including ties. SFT trains the student directly on reference solutions; OPD + SFT and MetaOPD + SFT add the SFT loss to the student objective at the indicated coefficient, with $\gamma^\star \approx 2.16$.

optimizer mini-batches of 32, and 696 student updates. MetaOPD performs a meta-update at each of the 696 student updates; OPD has no outer objective. The MetaOPD timing uses response-centered scores and the coupled schedules in Appendix B.4.

Training wall time is multiplied by all eight allocated GPUs, including the four paired teacher GPUs. Evaluation and diagnostic probes are excluded. The resulting 45.0 and 88.5 GPU-hours give a $1.97\times$ ratio for this 1.7B setting. These measurements quantify the cost of the complete training procedure, including the outer objective, rather than the small weighting network alone.

Table 10: Training-only cost for the 1.7B student on eight NVIDIA A100 80GB GPUs, including four student and four teacher GPUs. Both methods complete 696 student updates; MetaOPD performs one meta update per step. Evaluation and diagnostic probes are excluded.

Method	Training time (h)	GPU-hours	Relative cost
OPD	5.62	45.0	$1.00\times$
MetaOPD	11.06	88.5	$1.97\times$

C Motivating One-Step Comparison

C.1 Shared Setting

The comparison in Figure 1 uses Qwen3-1.7B-Base as the student and Qwen3-8B as the frozen teacher. Within each panel, all strategies start from the same initial student and optimizer state and perform one actual student update on the same student-generated responses. The optimizer starts with zero AdamW moments and completed-step count zero, uses a learning rate of 3×10^{-6}, and clips the global gradient norm at 1.0. Virtual and actual student updates use the same AdamW update rule, with the virtual update leaving the shared starting state unchanged. The completion mask retains the valid sampled EOS action.

We use four fixed prompt folds and three response-sampling seeds, giving 12 panels. Each panel contains 16 prompts with two responses per prompt. The seeds change only response sampling; the prompt selection and fold assignment remain fixed across seeds. We also sample two independent

reference sets, A and B, from the MATH training pool, each containing 64 prompts and their reference solutions, and share both sets across all panels.

C.2 Compared Strategies

Uniform OPD assigns equal weights to valid response tokens. MetaOPD starts from a weighting network initialized to produce uniform weights. It uses reference set A to compute the validation loss after a virtual weighted OPD update, performs one meta-update of the network, and recomputes the token weights. These weights then determine the actual student update from the original student parameters and optimizer state. Reference set B evaluates the resulting student and does not enter weight learning.

EOPD (Jin et al., 2026) and TIP (Xu et al., 2026b) apply their supervision-allocation rules within the shared sampled-OPD framework. TIP uses the Soft-OR selection described in Appendix B.2.1. Random assigns random nonuniform token weights, providing a reference for allocation without validation-loss feedback. All candidate student updates use the shared responses, learning rate, and clipping threshold described above.

C.3 Evaluation and Aggregation

Let θ'_h be the student after the actual update under strategy h. We evaluate its teacher-forced reference-solution loss on set B, averaged over valid reference tokens, and report the paired reduction relative to Uniform OPD:

$$R_h = \mathcal{L}_{\text{out}}(\theta'_{\text{Uniform}}; B) - \mathcal{L}_{\text{out}}(\theta'_h; B). \tag{29}$$

Positive R_h means that strategy h yields a lower post-update validation loss than Uniform OPD, matching the direction in Figure 1(b). We average the paired reductions across the 12 panels and obtain 95% percentile confidence intervals from 10,000 bootstrap resamples of panels, retaining the pairing between strategies. These intervals are conditional on the fixed prompts and reference sets; the response-sampling seeds are not independent full training runs.

In this comparison, MetaOPD achieves the largest observed reduction, EOPD yields a smaller improvement, and TIP and Random remain close to Uniform OPD. The result supports using post-update performance to learn token weights in this one-step setting; it does not establish the effects of repeatedly applying the strategies over a full training run.

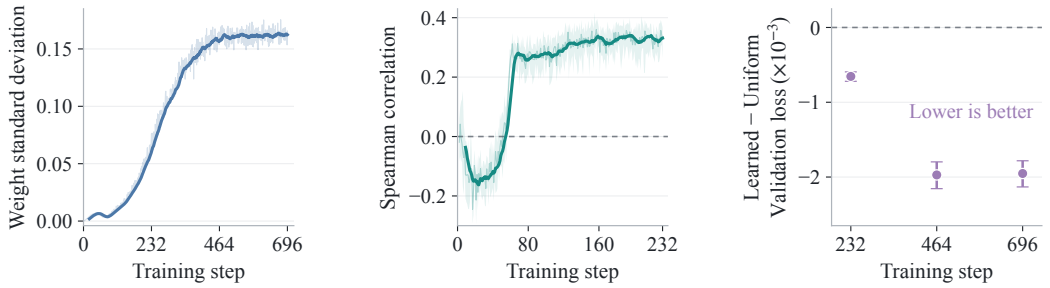
D Training Dynamics and Update Analysis

D.1 Diagnostic Setting

We use online traces to track token-weight evolution and fixed-checkpoint probes to examine agreement of loss derivatives across batches, learned allocation, and one-step update quality. Here $c_{i,t}$ is proportional to the negative validation-loss derivative with respect to a token weight, as defined in Equation (15). Figure 4(a) shows the weight standard deviation recorded during training. Figure 4(b) uses a separate, matched diagnostic reproduction with the full three-epoch, 696-step learning-rate schedule, recording only steps 1–232. It reports the median and interquartile range (IQR) of within-response Spearman correlations between pre-update token weights and $c_{i,t}$ computed using the current training reference batch. Light traces show per-step statistics; dark curves use complete trailing 20-step and 8-step arithmetic means in Figures 4(a) and 4(b), respectively. Constant initial weights make correlation undefined, so that point is left blank.

The checkpoint probes examine Qwen3-1.7B-Base trained with MetaOPD and the restricted-input variants of Section 4.3. Early, middle, and final denote MetaOPD steps 232, 464, and 696. The probes use each checkpoint's student AdamW state, with global gradient-clipping threshold 1.0. All candidates use the same AdamW update rule as training at reference learning rate 3×10^{-6}, independent of the decayed training rate, and are not committed to the model.

For the checkpoint probes, student-generated responses provide the data for candidate OPD updates, while reference solutions are used to compute validation loss. Checkpoints share the same rollout prompts but generate responses separately. Each checkpoint's panel contains 64 responses, divided



(a) Weight dispersion (b) Weight alignment (c) Update quality

Figure 4: **Training dynamics and update quality (1.7B).** (a) Token-weight standard deviation over training. (b) Median within-response Spearman correlation between pre-update token weights and $c_{i,t}$, the scaled negative validation-loss derivative, in epoch 1 of a matched diagnostic run; shading shows the interquartile range. (c) Paired validation loss difference $\Delta L_{\text{Learned}-\text{Uniform}}$ after one update from identical starting parameters at each of three checkpoints; negative values favor Learned. Error bars show 95% reference-bank bootstrap intervals with checkpoints and rollouts fixed. Appendix D gives the full protocol.

into two groups of 32; candidate updates are computed separately for each group from that checkpoint. All candidate weighting rules and reference banks at a given checkpoint reuse these responses. We also use 256 separate prompt–reference-solution pairs, partitioned into 32 reference banks of eight examples each. The rollout prompts and reference examples are drawn from MATH test with MATH500 excluded. Within each reference bank, validation loss is averaged over valid reference-solution tokens.

We compute the raw $c_{i,t}$ at the weights produced by the checkpoint's weighting network and define the centered diagnostic quantity

$$c^{\circ}_{i,t} = c_{i,t} - \frac{1}{n_i} \sum_{s} m_{i,s} c_{i,s}, \tag{30}$$

where $n_i = \sum_s m_{i,s} > 0$, using the ordinary mean over valid tokens within each response. The raw $c_{i,t}$ retains the derivative meaning in Equation (15); $c^{\circ}_{i,t}$ is a centered statistic, distinct from the weighted centering term in the derivative with respect to unnormalized weights. The diagnostic sign and magnitude comparisons below use $c^{\circ}_{i,t}$ and $|c^{\circ}_{i,t}|$. Subtracting a response-wise constant does not change within-response Spearman correlations or ranks.

To assess consistency across reference batches, we combine the 32 banks into eight disjoint batches of 32 reference examples. For each batch, we average the vectors $\boldsymbol{d}$ from its four banks, weighted by their valid reference-token counts, then compute and center $c_{i,t}$. We compare the resulting $c^{\circ}_{i,t}$ values for all pairs of batches within each response using Spearman correlation, sign agreement, and top-20% Jaccard overlap, then take the median over pairs and responses.

To assess learned-weight alignment, we average $\boldsymbol{d}$ across all 32 reference banks, weighted by their valid reference-token counts, using all 256 reference examples. We report the median within-response correlations of learned weights with the resulting $c^{\circ}_{i,t}$ values or their magnitudes. The 95% percentile intervals use 2,000 prompt-cluster bootstrap resamples, retaining all observations for each sampled prompt. This token-count weighting defines the derivative aggregation; the one-step loss comparisons below instead average banks equally.

D.2 Learned Weights and Update Quality

In the 1.7B setting, weight dispersion increases from uniform initialization and then levels off (Figure 4(a)), while each response's coefficient sum remains fixed. In the matched diagnostic run, the online correlation between token weights and $c_{i,t}$ turns from negative to positive and reaches about 0.3 later in the recorded first epoch (Figure 4(b)).

The online curve uses the current training reference batch, whereas checkpoint alignment uses that checkpoint's rollout panel and $c^{\circ}_{i,t}$ computed from the averaged vector over 256 reference examples, so their correlations are not interchangeable.

The $c^{\circ}_{i,t}$ values agree across independent reference batches at all three checkpoints (rank correlation around 0.98; sign agreement above 0.99; Table 11a), providing a reproducible basis for analyzing the learned weights.

Learned weights correlate positively with $c^{\circ}_{i,t}$ at all three checkpoints, with 95% bootstrap intervals above zero (Table 11b); the final correlation is 0.267 $[0.246, 0.278]$. The intervals for correlations with $|c^{\circ}_{i,t}|$ include zero, leaving that association inconclusive. Because the student and optimizer states change, the derivatives used for comparison also change; these correlations therefore do not establish convergence of the weighting rule.

Table 11: Loss-derivative agreement and learned-weight correlations for MetaOPD at three training stages (1.7B). Agreement compares $c^{\circ}_{i,t}$ computed from 32-example reference batches; weight correlations use $c^{\circ}_{i,t}$ obtained after averaging $\boldsymbol{d}$ over 256 reference examples, weighted by valid reference-token counts, and centering within each response. Values summarize within-response comparisons. Brackets report 95% intervals from 2,000 prompt-cluster bootstrap resamples.

(a) Agreement across independent reference batches

Checkpoint	Student steps	Spearman	Sign agreement	Top-20% Jaccard
Early	232	0.975	0.993	0.958
Middle	464	0.976	0.993	0.961
Final	696	0.975	0.993	0.953

(b) Weight alignment: Spearman correlation [95% CI]

Checkpoint	$c^{\circ}_{i,t}$	$\lvert c^{\circ}_{i,t}\rvert$
Early	0.271 $[0.249, 0.286]$	0.004 $[-0.116, 0.169]$
Middle	0.266 $[0.255, 0.284]$	0.081 $[-0.093, 0.190]$
Final	0.267 $[0.246, 0.278]$	0.082 $[-0.064, 0.204]$

At the final checkpoint, correlations with $c^{\circ}_{i,t}$ are 0.007 for position, 0.216 for teacher entropy, and -0.569 for the teacher–student log-probability gap. The gap has a larger absolute correlation than the learned weights (0.267), in the opposite direction. These within-response correlations do not directly measure validation loss after a student update.

D.3 One-Step Update Comparisons

We compare Uniform, Learned, and Reweighted allocations from the same checkpoint, computing candidate updates separately for each of the two rollout groups. Learned uses weights directly from the checkpoint's frozen weighting network, without a new meta-update; this probe evaluates the allocation already learned at that checkpoint. Reweighted multiplies these learned weights by a linear factor in $[0.9, 1.1]$ based on the within-response rank of $c^{\circ}_{i,t}$, then restores unit response mean. It uses the same eight-example reference bank to compute the ranking and evaluate the update, so this comparison gives Reweighted access to its evaluation objective.

Let B_b denote reference bank b and $\theta'_{h,b,g}$ the candidate for strategy h, bank b, and rollout group g. Within each bank, we compute token-mean validation loss and average the two group results equally, then average the 32 banks equally:

$$\Delta L_h^{\text{step}} = \frac{1}{32}\sum_{b=1}^{32}\frac{1}{2}\sum_{g=1}^{2}\left[\mathcal{L}_{\text{out}}(\theta'_{h,b,g}; B_b) - \mathcal{L}_{\text{out}}(\theta; B_b)\right], \tag{31}$$

where $h \in \{\text{Uniform}, \text{Learned}, \text{Reweighted}\}$. Negative ΔL_h^{step} indicates a lower mean loss than the unchanged checkpoint. Uniform and Learned do not use the bank to construct their weights, so their candidates depend only on the rollout group. The paired contrasts are

$$\Delta L_{\text{Learned}-\text{Uniform}} = \Delta L^{\text{step}}_{\text{Learned}} - \Delta L^{\text{step}}_{\text{Uniform}}, \tag{32}$$

$$\Delta L_{\text{Reweighted}-\text{Learned}} = \Delta L^{\text{step}}_{\text{Reweighted}} - \Delta L^{\text{step}}_{\text{Learned}}. \tag{33}$$

The pre-update losses cancel because each contrast uses the same banks and rollout groups.

For Learned–Uniform (Figure 4(c); Table 12), we average the two paired group differences within each bank, then resample the 32 banks with replacement 10,000 times and recompute their equally weighted mean to obtain a 95% percentile interval. These intervals condition on the checkpoint and its fixed responses, excluding training-seed variation.

At the diagnostic reference learning rate of 3×10^{-6}, Learned yields lower validation loss than Uniform at all three checkpoints (Figure 4(c) and Table 12), with a final paired difference of approximately -1.95×10^{-3}. Both allocations nevertheless increase mean loss relative to the unchanged checkpoint at every stage, so Learned's advantage is relative rather than absolute.

Reweighted yields lower validation loss than Learned at each checkpoint, although all three allocations increase loss at the final checkpoint (Table 12). These interventions use local derivative rankings to take finite steps at the reference rate; their relative ordering does not guarantee descent or improvement over the full training trajectory.

Table 12: One-step validation loss changes $\Delta L_h^{\mathrm{step}}$ and paired contrasts (10^3 scale; lower is better) at reference learning rate 3×10^{-6}. Learned$-$Uniform intervals use 10,000 reference-bank bootstrap resamples with checkpoint and rollouts fixed. Reweighted uses the evaluation bank to construct weights.

Checkpoint (step)	*(a) Validation loss changes*			*(b) Paired contrasts*		
	Uniform	Learned	Reweighted	Learned$-$Uniform	95% CI	Reweighted$-$Learned
Early (232)	$+0.74$	$+0.09$	-3.32	-0.65	$[-0.72, -0.59]$	-3.41
Middle (464)	$+2.78$	$+0.80$	-3.03	-1.97	$[-2.16, -1.80]$	-3.83
Final (696)	$+6.26$	$+4.31$	$+0.61$	-1.95	$[-2.13, -1.78]$	-3.70

Restricted-input allocations. All four restricted-input variants are evaluated at student step 696 and yield consistent derivative estimates and negative Reweighted–Learned contrasts under the same protocol (Table 13), assessing local update quality rather than benchmark accuracy.

Table 13: Loss-derivative agreement and update comparisons for restricted-input variants (1.7B), all at student step 696. Agreement uses $c_{i,t}^{\circ}$ computed from 32-example reference batches; $\Delta L_{\mathrm{Reweighted-Learned}}$ uses equally weighted eight-example reference banks at reference rate 3×10^{-6}. Point estimates; negative contrasts favor Reweighted.

Input restriction	Spearman	Sign agreement	Top-20% Jaccard	$10^3 \Delta L_{\mathrm{Reweighted-Learned}}$
Position only	0.978	0.994	0.962	-3.56
Teacher entropy	0.978	0.994	0.961	-3.92
Log-prob. gap	0.979	0.993	0.959	-3.84
Top-32 profiles	0.975	0.992	0.960	-3.30

E Out-of-Domain Generalization

We evaluate the Qwen3-1.7B-Base checkpoints from the MATH main experiment on GPQA-Diamond (Rein et al., 2023), MMLU-Pro (Wang et al., 2024), and AlpacaEval 2.0 (Dubois et al., 2024), without additional training on these tasks. Distillation methods use the Qwen3-8B teacher. All entries in Table 2 are obtained from local evaluation under a common protocol for each benchmark; Base Model denotes Qwen3-1.7B-Base before MATH training.

Evaluation protocol. Across these benchmarks, we use temperature 1.0, top-p 0.8, and a maximum response length of 8192 tokens, matching the underlying decoding settings in Table 5. The number of responses and evaluation metric are benchmark-specific.

GPQA-Diamond contains 198 questions, for which we sample eight responses per question. Avg@8 measures average response accuracy, and Pass@8 measures the fraction of questions answered correctly at least once. MMLU-Pro contains 12,032 test questions and uses five-shot chain-of-thought demonstrations matched to the question's category. We generate one response per question and report accuracy as Pass@1. AlpacaEval 2.0 compares model responses with GPT-4-Turbo reference responses using the `weighted_alpaca_eval_gpt4_turbo` annotator. We report both raw win rate (WR) and length-controlled win rate (LC-WR); these are preference-based scores, rather than answer-verification accuracy.

Aggregation. All scores are on a 0–100 scale. The Avg. row is the unweighted arithmetic average of the five displayed metric scores:

$$\begin{aligned}\mathrm{Avg} = \tfrac{1}{5}\big(&\mathrm{GPQA}_{\mathrm{Avg@8}} + \mathrm{GPQA}_{\mathrm{Pass@8}} + \mathrm{MMLU}_{\mathrm{Pass@1}} \\ &+ \mathrm{Alpaca}_{\mathrm{LC\text{-}WR}} + \mathrm{Alpaca}_{\mathrm{WR}}\big).\end{aligned}$$

This descriptive summary assigns 40%, 20%, and 40% of its weight to GPQA-Diamond, MMLU-Pro, and AlpacaEval, respectively. It is distinct from the six-benchmark mathematical-reasoning averages in Table 1; individual metrics remain necessary to interpret the transfer profile.

Results. MetaOPD has the highest reported scores on both GPQA-Diamond metrics and MMLU-Pro Pass@1. Its AlpacaEval scores are close to OPD's and below TIP's, showing that the observed advantage varies across the evaluated tasks. MetaOPD also has the highest descriptive average across the five reported metrics.