STAGEOPD: STAGE-AWARE LOSS REWEIGHTING FOR ON-POLICY DISTILLATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Current on-policy distillation (OPD) methods mainly improve how teacher supervision is collected on student-generated trajectories, but typically aggregate the resulting token-level losses with uniform or fixed token weights. In long reasoning trajectories, such token-level aggregation favors long routine reasoning spans, while short but informative regions, such as correction boundaries, teacher-guided spans, and the student's immediate continuation after guidance, receive much less optimization weight simply because they contain fewer tokens. We introduce STAGEOPD, a simple stage-aware loss-reweighting method for on-policy distillation. STAGEOPD decomposes each trajectory into routine student reasoning, correction boundaries, teacher-guided spans, and post-guidance recovery, assigns an explicit loss budget to each stage, and normalizes the resulting weights within each sequence so that the total optimization weight of a stage is determined by its role rather than its token count, without requiring additional rollout or teacher computation. Experiments with a Qwen3-1.7B-Non-Thinking student and a Qwen3-4B-Instruct-2507 teacher on DAPO-Math-17K show that STAGEOPD achieves 45.58 ± 0.08 average accuracy across eight mathematical reasoning benchmarks, outperforming the strongest baseline by 0.11 percentage points. Ablation studies further show that removing the student-reasoning anchor, the teacher-guidance budget, or sequence-level weight normalization reduces average accuracy by 1.98, 1.68, and 1.08 points, respectively; a Recovery-window study further achieves its best result at 64 tokens. These results demonstrate the effectiveness of stage-aware loss allocation for on-policy distillation.

1 INTRODUCTION

On-policy distillation (OPD) has emerged as an effective paradigm for improving language models by training the student on states induced by its own policy rather than relying solely on static teacher-generated data (Song & Zheng, 2026; Zhao et al., 2026; Heo et al., 2026). This property is particularly important for long-form reasoning, where the student's intermediate decisions continuously shape the states on which subsequent supervision is received. However, an important optimization problem remains underexplored: once token-level supervision has been obtained along a long reasoning trajectory, how should different parts of that trajectory contribute to learning? Standard token-wise aggregation treats tokens as the basic units of optimization. Consequently, long spans of routine student reasoning can dominate the training objective simply because they contain many tokens, while shorter but informative regions receive only a small fraction of the total optimization weight.

Recent OPD studies have substantially improved both the form and the placement of on-policy supervision. Existing work explores improved distillation objectives, representation-level supervision, unified formulations, and dual-perspective training strategies (Heo et al., 2026; Yang et al., 2026b; Hou et al., 2026; Yu et al., 2026). Other approaches extend on-policy learning through reward-based generalization and self-distillation (Yang et al., 2026c;a). More recently, Trajectory-Relayed On-Policy Distillation introduces local teacher interventions into student-generated reasoning trajectories, enabling targeted teacher guidance during generation (Xu et al., 2026). These advances improve what supervision is provided and where it enters the trajectory. Yet the resulting token-level losses are still typically aggregated according to token frequency or fixed token weights, leaving the total influence of each region strongly tied to how many tokens it contains.

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.

This limitation becomes particularly pronounced in long reasoning trajectories. Routine student reasoning may span thousands of tokens, whereas a teacher-intervention boundary can consist of only a single token, a teacher-guided span is typically much shorter, and the student's immediate continuation after guidance occupies only a limited local window. Increasing the fixed weight of these tokens can make each individual token more influential, but it does not directly control how much the corresponding region contributes as a whole. In our training traces, for example, the teacher-intervention boundary accounts for only 0.0232% of valid tokens and receives only 0.0909% of the total weight even with a four-fold token weight. Thus, token-level reweighting alone does not provide direct control over the optimization contribution of structurally different parts of a reasoning trajectory.

To address this problem, we introduce STAGEOPD, a simple *stage-aware loss reweighting* method for on-policy distillation. STAGEOPD organizes a locally guided trajectory into four structural stages: routine student reasoning (NORMAL), the teacher-intervention boundary (TRIGGER), the teacher-guided span (TEACHER BODY), and the student's immediate continuation after teacher guidance (RECOVERY). Instead of assigning importance to individual tokens first, STAGEOPD assigns an explicit optimization budget to each stage present in a response and then distributes that budget across the tokens within the stage. A sequence-level normalization step preserves the overall loss scale. As a result, the relative contribution of a stage is controlled by its assigned budget rather than by its token count.

The proposed method changes only the loss aggregation procedure and requires no additional rollout, teacher inference, reward model, or auxiliary supervision. It can therefore be directly applied after the standard token-level OPD loss has been computed and is complementary to advances in supervision design and trajectory construction. Figure 1 illustrates the central distinction: fixed token weights determine stage-level influence indirectly through token count, whereas STAGEOPD first allocates optimization weight at the stage level and then distributes it among the corresponding tokens.

We evaluate STAGEOPD with a Qwen3-1.7B-Non-Thinking student and a Qwen3-4B-Instruct-2507 teacher (Yang et al., 2025), trained on DAPO-Math-17K and evaluated across eight mathematical reasoning benchmarks. STAGEOPD achieves an average accuracy of **45.58±0.08**, exceeding the strongest baseline by average accuracy by 0.11 percentage points. A direct allocation audit shows that the three short stages together account for less than 2.4% of tokens while receiving about 18.9% of the optimization weight. Component ablations consistently degrade performance, with the largest drop reaching 1.98 points, while the Recovery-window analysis further supports explicitly modeling the student's immediate continuation after teacher guidance. Together, these results demonstrate the effectiveness of stage-aware loss allocation for on-policy distillation.

Our contributions are threefold:

1. We identify an overlooked limitation of token-wise loss aggregation in OPD: even with fixed token reweighting, the total optimization contribution of a structural region remains strongly coupled to its token count.
2. We introduce STAGEOPD, a simple stage-aware loss reweighting method that explicitly allocates optimization weight across structural stages and then distributes it within each stage, while preserving the overall sequence-level loss scale and requiring no additional model computation.
3. We demonstrate the effectiveness of stage-aware loss allocation across eight mathematical reasoning benchmarks, supported by direct allocation measurements, component ablations, and Recovery-window analysis.

2 RELATED WORK

On-policy distillation for long-form reasoning. Knowledge distillation transfers teacher supervision through predictions or generated targets (Hinton et al., 2015), while on-policy distillation (OPD) learns from student-induced states (Song & Zheng, 2026). Recent work explores delta-based objectives (Heo et al., 2026), representation distillation (Yang et al., 2026b), unified and dual formulations (Hou et al., 2026; Yu et al., 2026), reward extrapolation (Yang et al., 2026c), and self-distillation (Zhao et al., 2026; Yang et al., 2026a), increasingly targeting mathematical, scientific, and code reasoning (Yang et al., 2025; Shao et al., 2024; Moshkov et al., 2025; Team et al., 2026; Hendrycks

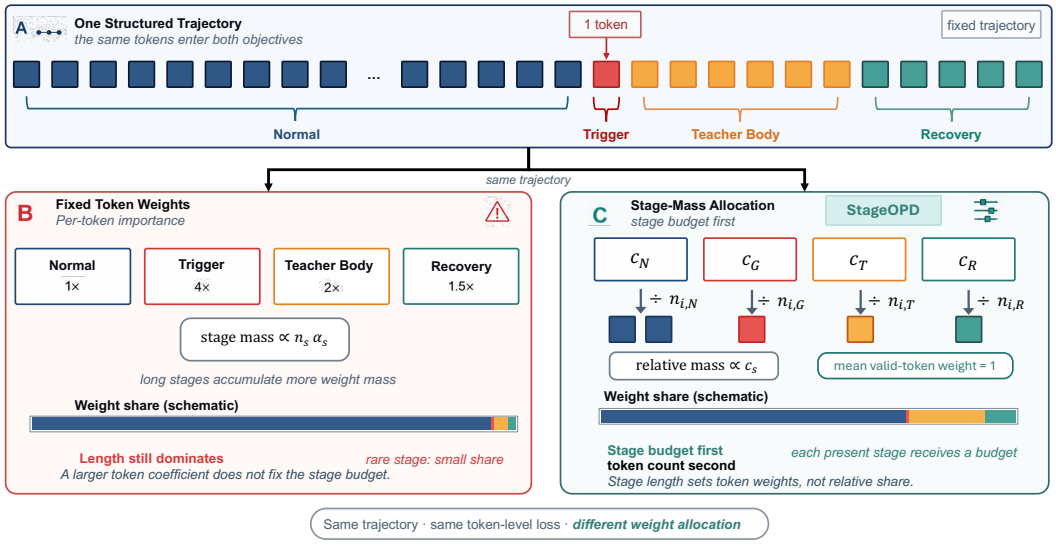


Figure 1: **Stage-aware loss allocation in StageOPD.** Both objectives operate on the same structured trajectory and token-level distillation losses. Fixed token weights determine the influence of each stage indirectly through its token count, whereas StageOPD first assigns an explicit budget to each stage and then distributes that budget among its tokens. Sequence-level normalization preserves the overall loss scale.

et al., 2021; Rein et al., 2023; Wang et al., 2023; Du et al., 2026; Jain et al., 2025). Our focus is complementary: how token-level losses from different trajectory regions should contribute to optimization.

Structured trajectories and localized supervision. Trajectory-Relayed On-Policy Distillation interleaves student reasoning with local teacher interventions (Xu et al., 2026), while reasoning-and-acting methods incorporate tools and environments (Yao et al., 2022b; Jin et al., 2025; Shridhar et al., 2020; Yao et al., 2022a). Agent research further studies long-horizon behavior, automated research, data-efficient agency, dataset discovery, and resource-aware evaluation (Li et al., 2026c; Jiang et al., 2026; Wu et al., 2026b; Fu et al., 2026; Xiao et al., 2025; Li et al., 2025; 2026a;b). StageOPD uses the resulting structural boundaries to reallocate optimization weight without altering trajectory construction.

Loss weighting and credit-aware on-policy learning. Loss allocation is closely related to credit assignment in policy optimization (Schulman et al., 2017). Recent structured approaches include StepOPSD (Zhang et al., 2026), UCOB (Tu et al., 2026), SEED and self-distilled agentic reinforcement learning (Wu et al., 2026a; Lu et al., 2026), and group-in-group policy optimization (Feng et al., 2026). StageOPD instead assigns each structural stage an explicit budget, distributes it across stage tokens, and normalizes weights at the sequence level, decoupling stage contribution from token frequency without additional rollout or scoring models.

3 StageOPD

3.1 Structured OPD trajectories

For a prompt x_i, the student policy π_θ generates a response

$$y_i = (y_{i,1}, \ldots, y_{i,N_i}), \tag{1}$$

where N_i is the valid response length. We consider structured OPD trajectories that may contain local teacher interventions, following the trajectory construction studied in Xu et al. (2026). The exact controller used in our implementation is described in Section A.1.

Table 1: **Four structural stages used by STAGEOPD.** The stages are defined by token source and position in the trajectory. RECOVERY contains at most the first r student tokens after a teacher span; all remaining valid student tokens are NORMAL.

Stage	Symbol	Structural definition	Training role
Normal	N	Student token outside Recovery	Preserve the student's on-policy reasoning and core capability.
Trigger	G	First token of each teacher span	Mark the boundary where teacher guidance begins.
Teacher Body	T	Remaining tokens in a teacher span	Provide local teacher guidance under the visited prefix.
Recovery	R	First r student tokens after a teacher span	Train the student to continue autonomously after teacher guidance.

Let

$$R_{i,t} \in \{0,1\}, \qquad T_{i,t} \in \{0,1\} \tag{2}$$

denote the valid-response mask and teacher-token mask, respectively. The current student π_θ, a frozen behavior policy $\pi_{\bar{\theta}}$, and the teacher π_T provide the quantities required to compute the clipped per-token distillation loss $\ell_{i,t}$. STAGEOPD operates after these token-level losses have been obtained and changes only how they are aggregated over the trajectory.

A locally guided trajectory naturally contains structurally different regions. We partition every valid response position into the four mutually exclusive stages summarized in Table 1: NORMAL, TRIGGER, TEACHER BODY, and RECOVERY. The first token of each teacher span is TRIGGER, the remaining teacher tokens form TEACHER BODY, and the first r student tokens following a teacher span form RECOVERY. All remaining valid student tokens belong to NORMAL.

If a response contains no teacher intervention, only NORMAL is present. If multiple interventions occur, RECOVERY restarts after each teacher span and terminates when its window ends or a new teacher span begins.

3.2 LIMITATION OF TOKEN-WISE LOSS WEIGHTING

Standard token-wise aggregation assigns optimization weight at the token level. A fixed-weight variant can be written as

$$\mathcal{L}_{\text{token}} = \frac{\sum_t \alpha_{s(t)} \ell_t}{\sum_t \alpha_{s(t)}}, \tag{3}$$

where $s(t)$ denotes the structural stage containing token t and α_s is the corresponding fixed token weight.

For a stage containing n_s tokens, its total unnormalized optimization weight is proportional to $n_s \alpha_s$. Increasing α_s therefore increases the importance of each token in that stage, but the stage's total contribution remains directly coupled to its token count. This is precisely the limitation highlighted in the Introduction: token-level weighting does not directly control how much a structural stage contributes as a whole.

Table 2: **Optimization-weight allocation under fixed token weights.** Weight share is obtained by normalizing the product of token frequency and the corresponding fixed token weight. Long stages therefore accumulate most of the total optimization weight.

Stage	Token share (%)	Token weight	Weight share (%)
Normal	97.3838	1.0	95.2619
Trigger	0.0232	4.0	0.0909
Teacher Body	1.7225	2.0	3.3699
Recovery	0.8705	1.5	1.2773

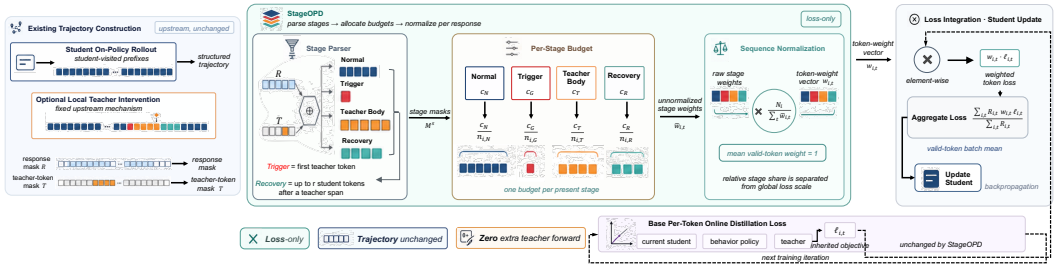


Figure 2: **StageOPD at the loss-aggregation interface.** The response and teacher-token masks define four structural stages. StageOPD first assigns an optimization budget to each present stage, distributes the budget across the corresponding tokens, and normalizes the resulting weights at the sequence level. These weights are applied to the unchanged token-level distillation loss before aggregation. The upstream trajectory construction and teacher computation remain unchanged.

The fixed-token comparison uses

$$(\alpha_N, \alpha_G, \alpha_T, \alpha_R) = (1.0, 4.0, 2.0, 1.5). \tag{4}$$

As shown in Table 2, Normal accounts for 97.3838% of valid tokens and consequently receives 95.2619% of the total optimization weight. In contrast, Trigger accounts for only 0.0232% of tokens and receives only 0.0909% of the total weight despite its $4\times$ coefficient. The short Teacher Body and Recovery stages are similarly limited by their token frequency.

3.3 Stage-aware loss reweighting

Stage masks. We first construct the four stage masks from R and T. Define $\mathrm{shift}(T)_{i,t} = T_{i,t-1}$, with zero padding at the first response position. The Trigger and Teacher Body masks are

$$M^G_{i,t} = T_{i,t} \wedge \neg T_{i,t-1}, \tag{5}$$

$$M^T_{i,t} = T_{i,t} \wedge \neg M^G_{i,t}. \tag{6}$$

For each teacher-span endpoint, the next at most r student positions form M^R. The remaining valid student positions form

$$M^N_{i,t} = (R_{i,t} \wedge \neg T_{i,t}) \wedge \neg M^R_{i,t}. \tag{7}$$

The four masks are pairwise disjoint and cover every valid response position:

$$M^N + M^G + M^T + M^R = R. \tag{8}$$

Our main configuration uses a Recovery window of $r = 64$ student tokens.

Stage-level allocation. For each stage $s \in \{N, G, T, R\}$, let $c_s \geq 0$ denote its stage budget and

$$n_{i,s} = \sum_t M^s_{i,t} \tag{9}$$

Table 3: **Main stage budgets used by StageOPD.** The final column gives the corresponding relative optimization-weight share when all four stages are present.

Stage	Budget c_s	Weight share (%)
Normal	1.00	73.529
Trigger	0.01	0.735
Teacher Body	0.25	18.382
Recovery	0.10	7.353

denote its token count in response i. Rather than assigning a fixed weight independently to every token, StageOPD first assigns budget c_s to stage s and distributes it uniformly across the tokens belonging to that stage:

$$\widetilde{w}_{i,t} = \sum_s M^s_{i,t} \frac{c_s}{\max(n_{i,s}, 1)}. \tag{10}$$

We then normalize the weights within each response:

$$w_{i,t} = N_i \frac{\widetilde{w}_{i,t}}{\sum_{u:R_{i,u}=1} \widetilde{w}_{i,u}}, \tag{11}$$

where

$$N_i = \sum_t R_{i,t}. \tag{12}$$

For a nonempty response whose present stages contain at least one positive budget, the denominator in Equation (11) is positive. The main configuration assigns positive budgets to all four stages.

The resulting training objective is

$$\mathcal{L}_{\text{StageOPD}} = \frac{\sum_i \sum_t R_{i,t} w_{i,t} \ell_{i,t}}{\sum_i N_i}. \tag{13}$$

The underlying token-level distillation loss $\ell_{i,t}$ is unchanged. In our implementation, it is computed before StageOPD using the existing clipped policy-gradient objective (Schulman et al., 2017); the complete inherited objective is given in Section A.1.

3.4 Properties of stage-aware allocation

The allocation in Equations (10) and (11) has two useful properties.

Stage-level relative allocation. Let

$$\mathcal{S}_i^+ = \{s : n_{i,s} > 0\} \tag{14}$$

be the set of stages present in response i. If $\sum_{u \in \mathcal{S}_i^+} c_u > 0$, then for any $s \in \mathcal{S}_i^+$,

$$\sum_t M^s_{i,t} w_{i,t} = N_i \frac{c_s}{\sum_{u \in \mathcal{S}_i^+} c_u}. \tag{15}$$

Therefore, the relative optimization-weight share assigned to a present stage is

$$\frac{c_s}{\sum_{u \in \mathcal{S}_i^+} c_u}, \tag{16}$$

which is independent of the stage token count $n_{i,s}$. Stage length determines how the stage budget is distributed among its tokens, but not the stage's relative share of the response-level optimization weight.

Sequence-level scale preservation. By construction,

$$\sum_t R_{i,t} w_{i,t} = N_i, \qquad \frac{1}{N_i} \sum_t R_{i,t} w_{i,t} = 1. \tag{17}$$

Thus, changing the relative stage budgets does not change the mean valid-token weight of a response. Stage-level allocation and the overall loss scale are controlled separately.

3.5 Main configuration and optimization view

Our main stage-budget configuration is

$$(c_N, c_G, c_T, c_R) = (1.00, 0.01, 0.25, 0.10). \tag{18}$$

When all four stages are present, these budgets produce the relative allocation shown in Table 3. If a stage is absent, it is excluded from the normalization in Equation (11). In particular, a response

Table 4: **Mathematical reasoning performance across eight benchmarks.** Avg denotes the unweighted mean over the eight benchmark accuracies defined in Equation (21). Bold and underline indicate the best and second-best results among the student-training methods, respectively.

Method	A24	A25	A26	M500	AMC23	Olymp.	HMMT Feb26	HMMT Nov25	Avg
Teacher (4B)	60.42	46.04	52.19	94.20	93.83	70.62	31.25	41.88	61.30
Student	12.60	9.58	7.40	71.95	47.89	38.54	6.34	4.38	24.84
SFT	23.33	19.48	16.15	81.40	59.45	46.62	12.59	6.56	33.20
KD	23.54	21.15	15.31	81.45	60.23	48.07	12.78	7.50	33.75
GRPO	24.58	22.08	15.62	80.35	60.16	48.74	14.49	9.38	34.42
OPD	35.83	25.52	23.33	85.70	70.08	55.27	20.08	14.06	41.23
TRD	19.27	19.69	12.71	77.70	55.47	44.18	11.93	4.58	30.69
FastOPD	**42.29**	<u>30.42</u>	26.35	87.95	74.30	58.16	**23.58**	**20.73**	<u>45.47</u>
SKD	33.12	**30.73**	<u>28.85</u>	87.35	72.42	54.41	20.08	11.88	42.35
SARD-FULL	<u>40.73</u>	27.92	26.35	<u>88.45</u>	<u>76.80</u>	<u>60.94</u>	<u>22.35</u>	<u>16.77</u>	45.04
STAGEOPD (ours)	38.02	28.33	**29.48**	**89.10**	**81.09**	**61.05**	21.21	16.35	**45.58**

without teacher intervention contains only NORMAL and therefore assigns all of its optimization weight to that stage.

The stage-level structure is also visible directly in the gradient. Define the per-response objective

$$\mathcal{L}_i = \frac{1}{N_i} \sum_t R_{i,t} w_{i,t} \ell_{i,t}, \qquad \mathcal{L}_{\text{STAGEOPD}} = \frac{\sum_i N_i \mathcal{L}_i}{\sum_i N_i}. \tag{19}$$

Conditioned on the sampled trajectory and its structural masks, $w_{i,t}$ is constant with respect to θ. Writing $g_{i,t} = \nabla_\theta \ell_{i,t}$ gives

$$\nabla_\theta \mathcal{L}_i = \sum_{s \in \mathcal{S}_i^+} \frac{c_s}{\sum_{u \in \mathcal{S}_i^+} c_u} \left(\frac{1}{n_{i,s}} \sum_{t \in s} g_{i,t} \right). \tag{20}$$

Each stage therefore first contributes its average token gradient, after which the stage budgets determine how these stage-level averages are combined.

3.6 INTEGRATION AND COMPLEXITY

STAGEOPD operates entirely at the loss-aggregation interface. As illustrated in Figure 2, the trajectory construction and token-level distillation objective are computed as usual. STAGEOPD then parses the structural stages, assigns stage budgets, distributes each budget within its stage, and normalizes the resulting token weights before loss aggregation.

For batch size B and response length N, stage-mask construction and weight computation require $O(BN)$ operations, the same asymptotic order as the token-level loss itself. The method introduces four Boolean stage masks, one token-weight tensor, and sequence-level reductions. It requires no additional rollout, sampling call, or teacher forward pass.

4 EXPERIMENTS

We evaluate STAGEOPD from three complementary perspectives. First, we compare its mathematical reasoning performance with supervised, reinforcement-learning, and on-policy distillation baselines. Second, we directly examine whether stage-aware loss reweighting produces the intended optimization-weight allocation across structural stages. Third, we study the contribution of the main design choices through component ablations and Recovery-window sensitivity.

Table 5: **Token frequency and optimization-weight allocation.** Weight share denotes the fraction of normalized optimization weight assigned to each stage. Weight/token ratio is the weight share divided by the corresponding token share.

Stage	Fixed token share (%)	Fixed weight share (%)	STAGEOPD token share (%)	STAGEOPD weight share (%)	Weight / token ratio
Normal	97.3838	95.2619	97.6207	81.1274	0.83×
Trigger	0.0232	0.0909	0.0216	0.5249	24.25×
Teacher Body	1.7225	3.3699	1.5468	13.1216	8.48×
Recovery	0.8705	1.2773	0.8109	5.2262	6.45×

4.1 EXPERIMENTAL SETUP

Models and Baselines. We use Qwen3-1.7B-Non-Thinking as the student and Qwen3-4B-Instruct-2507 as the teacher (Yang et al., 2025). Training uses the processed DAPO-Math-17K dataset. The main STAGEOPD configuration follows the stage budgets and Recovery window specified in Section 3.5. We compare against supervised distillation baselines including SFT and KD (Hinton et al., 2015), the outcome-reward method GRPO (Shao et al., 2024), and a broad set of on-policy and trajectory-based distillation methods. These include OPD, TRD, FastOPD, SKD, and the fixed-token structured OPD baseline SARD-FULL. The OPD family represents recent efforts to improve learning from student-generated trajectories (Song & Zheng, 2026; Heo et al., 2026), while trajectory-relayed distillation introduces localized teacher guidance during student generation (Xu et al., 2026). Student and Teacher results are included as capability references.

Evaluation. We evaluate on eight mathematical reasoning benchmarks: AIME 2024, AIME 2025, AIME 2026, MATH500 (Hendrycks et al., 2021), AMC 2023, OlympiadBench, HMMT February 2026, and HMMT November 2025. Sampling uses temperature 1.0, top-$p = 1.0$, and a maximum generation length of 32,768. We draw $K_b = 32$ samples per problem for AIME, AMC, and HMMT, and $K_b = 4$ for MATH500 and OlympiadBench. Let z_{bpk} indicate whether sample k for problem p on benchmark b is correct. We report benchmark accuracy and the unweighted average across all eight benchmarks:

$$\mathrm{Acc}_b = \frac{1}{P_b K_b} \sum_{p=1}^{P_b} \sum_{k=1}^{K_b} z_{bpk}, \qquad \mathrm{Avg} = \frac{1}{8} \sum_{b=1}^{8} \mathrm{Acc}_b \,. \tag{21}$$

4.2 MAIN RESULTS

Table 4 summarizes the overall benchmark performance. STAGEOPD achieves the highest average accuracy of **45.58**, improving over FastOPD, the strongest baseline by average accuracy, by 0.11 points. It also improves over the fixed-token SARD-FULL baseline by 0.54 points and over standard OPD by 4.35 points. The improvement is particularly clear on several challenging benchmarks. STAGEOPD achieves the best performance on four of the eight tasks: AIME 2026, MATH500, AMC 2023, and OlympiadBench. Compared with FastOPD, it improves AIME 2026 by 3.13 points,

Table 6: **Component ablations of STAGEOPD.** Δ denotes the change in eight-benchmark average accuracy relative to the full STAGEOPD configuration. Every component removal reduces performance.

Variant	Avg	Δ
STAGEOPD	**45.58**	–
Fixed-token SARD-FULL	45.04	−0.54
No Trigger budget	45.30	−0.28
No Teacher Body budget	43.90	−1.68
No Recovery budget	44.80	−0.78
No Normal budget	43.60	−1.98
No sequence-level normalization	44.50	−1.08

Table 7: **Sensitivity to Recovery-window length.** The main configuration with $r = 64$ achieves the highest average accuracy among the five tested window sizes.

Recovery tokens r	0	16	32	64	128
Avg accuracy	44.80	45.15	45.38	**45.58**	45.20
Δ vs. $r = 64$	-0.78	-0.43	-0.20	0.00	-0.38

MATH500 by 1.15 points, AMC 2023 by 6.79 points, and OlympiadBench by 2.89 points. These results show that stage-aware loss allocation provides consistent gains on several mathematical reasoning settings where maintaining useful intermediate guidance is important. We further evaluate run-to-run performance for the three strongest structured and on-policy baselines. Across three runs, STAGEOPD achieves 45.58 ± 0.08, compared with 45.47 ± 0.11 for FastOPD and 45.04 ± 0.12 for SARD-FULL. STAGEOPD therefore achieves both the highest mean accuracy and the lowest standard deviation among the three methods.

4.3 STAGE-AWARE ALLOCATION ANALYSIS

The central goal of STAGEOPD is to prevent short structural stages from being overwhelmed by long routine reasoning spans. We therefore directly measure the token frequency and optimization-weight share of each stage. The results in Table 5 show a clear separation between how often a stage appears and how much optimization weight it receives. Under fixed token weighting, the three short stages together receive only about 4.74% of the total optimization weight. With STAGEOPD, TRIGGER, TEACHER BODY, and RECOVERY occupy only about 2.38% of tokens but receive approximately 18.87% of the optimization weight. At the same time, NORMAL retains 81.13% of the total weight and remains the dominant optimization stage. The redistribution is particularly strong for the shortest stages. Relative to their token frequencies, TRIGGER, TEACHER BODY, and RECOVERY receive $24.25\times$, $8.48\times$, and $6.45\times$ as much optimization weight, respectively. These results directly confirm that stage-aware allocation prevents short but structurally important regions from disappearing under token-wise aggregation.

4.4 ABLATION AND RECOVERY-WINDOW ANALYSIS

We next examine how the main components of STAGEOPD contribute to performance. Table 6 removes one allocation component at a time while keeping the rest unchanged. All removals reduce average accuracy. The largest drops come from removing the NORMAL budget (-1.98 points) and TEACHER BODY budget (-1.68 points), highlighting the importance of both routine student reasoning and teacher-guided supervision. Removing sequence-level normalization reduces performance by 1.08 points, while RECOVERY contributes a 0.78-point improvement.

We further vary the Recovery-window length in Table 7. Performance rises from 44.80 without RECOVERY to 45.58 at $r = 64$, then falls to 45.20 at $r = 128$. This supports modeling RECOVERY as a localized post-guidance region: a moderate window captures immediate student continuation better than either omitting it or extending it into routine reasoning. Together, these results further support the stage-aware design of STAGEOPD.

5 DISCUSSION

Conclusion. We introduce STAGEOPD, a stage-aware loss reweighting method that decouples the optimization contribution of trajectory regions from their token counts. Across eight mathematical reasoning benchmarks, STAGEOPD achieves 45.58 ± 0.08 average accuracy and outperforms the strongest baseline. Notably, TRIGGER, TEACHER BODY, and RECOVERY comprise less than 2.4% of tokens but receive about 18.9% of the optimization weight.

Limitations and Future Work. STAGEOPD currently relies on fixed stage budgets and discrete boundaries. Future work could explore adaptive weighting, larger models, and broader long-horizon settings such as code generation, tool use, and multi-turn reasoning, as well as combinations with other OPD objectives and trajectory-construction strategies.

REFERENCES

Xeron Du, Yifan Yao, Kaijing Ma, Bingli Wang, Tianyu Zheng, Minghao Liu, Yiming Liang, Xiaolong Jin, Zhenlin Wei, Chujie Zheng, et al. Supergpqa: Scaling llm evaluation across 285 graduate disciplines. *Advances in Neural Information Processing Systems*, 38, 2026.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for llm agent training. *Advances in Neural Information Processing Systems*, 38:46375–46408, 2026.

Dayuan Fu, Yunze Wu, Xiaojie Cai, Lyumanshan Ye, Shijie Xia, Zhen Huang, Weiye Si, Tianze Xu, Jie Sun, Keyu Li, et al. Argo: Asynchronous rollout with human guidance for research agent optimization. 2026.

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.

Byeongho Heo, Jaehui Hwang, Sangdoo Yun, and Dongyoon Han. On-policy delta distillation. *arXiv preprint arXiv:2607.15161*, 2026.

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

Wenjin Hou, Shangpin Peng, Weinong Wang, Zheng Ruan, Yue Zhang, Zhenglin Zhou, Mingqi Gao, Yifei Chen, Kaiqi Wang, Hongming Yang, et al. Uni-opd: Unifying on-policy distillation with a dual-perspective recipe. *arXiv preprint arXiv:2605.03677*, 2026.

Naman Jain, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. In *International Conference on Learning Representations*, volume 2025, pp. 58791–58831, 2025.

Mohan Jiang, Dayuan Fu, Junhao Shi, Ji Zeng, Weiye Si, Keyu Li, Xuefeng Li, Yang Xiao, Wenjie Li, Dequan Wang, et al. davinci-agency: Unlocking long-horizon agency data-efficiently. *arXiv preprint arXiv:2602.02619*, 2026.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*, 2025.

Keyu Li, Mohan Jiang, Dayuan Fu, Yunze Wu, Xiangkun Hu, Dequan Wang, and Pengfei Liu. Datasetresearch: Benchmarking agent systems for demand-driven dataset discovery. *arXiv preprint arXiv:2508.06960*, 2025.

Keyu Li, Jin Gao, and Dequan Wang. Aligned agents, biased swarm: Measuring bias amplification in multi-agent systems. *arXiv preprint arXiv:2604.08963*, 2026a.

Keyu Li, Jin Gao, and Dequan Wang. The cost of knowing: A resource-aware protocol for benchmarking hallucination beyond static leaderboards. *arXiv preprint arXiv:2607.24063*, 2026b.

Keyu Li, Junhao Shi, Yang Xiao, Mohan Jiang, Jie Sun, Yunze Wu, Dayuan Fu, Shijie Xia, Xiaojie Cai, Tianze Xu, et al. Agencybench: Benchmarking the frontiers of autonomous agents in 1m-token real-world contexts. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7422–7440, 2026c.

Zhengxi Lu, Zhiyuan Yao, Zhuowen Han, Zi-Han Wang, Jinyang Wu, Qi Gu, Xunliang Cai, Weiming Lu, Jun Xiao, Yueting Zhuang, et al. Self-distilled agentic reinforcement learning. *arXiv preprint arXiv:2605.15155*, 2026.

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

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.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

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.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning. *arXiv preprint arXiv:2010.03768*, 2020.

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

Gemma Team, Sherif El Abd, Vaibhav Aggarwal, Robin Algayres, Alek Andreev, Olivier Bachem, Ian Ballantyne, Cormac Brick, Victor Cărbune, Michelle Casbon, et al. Gemma 4 technical report. *arXiv preprint arXiv:2607.02770*, 2026.

Songjun Tu, Chengdong Xu, Qichao Zhang, Yiwen Ma, Yaocheng Zhang, Linjing Li, Dong Li, Xiangyuan Lan, and Dongbin Zhao. Ucob: Learning to utilize and evolve agentic skills via credit-aware on-policy bidirectional self-distillation. *arXiv preprint arXiv:2606.29502*, 2026.

Xiaoxuan Wang, Ziniu Hu, Pan Lu, Yanqiao Zhu, Jieyu Zhang, Satyen Subramaniam, Arjun R Loomba, Shichang Zhang, Yizhou Sun, and Wei Wang. Scibench: Evaluating college-level scientific problem-solving abilities of large language models. *arXiv preprint arXiv:2307.10635*, 2023.

Jinyang Wu, Shuo Yang, Zhengxi Lu, Fan Zhang, Yuhao Shen, Lang Feng, Haoran Luo, Zheng Lian, Shuai Zhang, Zhengqi Wen, et al. Seed: Self-evolving on-policy distillation for agentic reinforcement learning. *arXiv preprint arXiv:2607.14777*, 2026a.

Yunze Wu, Dayuan Fu, Weiye Si, Zhen Huang, Mohan Jiang, Keyu Li, Shijie Xia, Jie Sun, Tianze Xu, Yang Xiao, et al. Innovatorbench: Evaluating agents' ability to conduct innovative ai research. In *International Conference on Learning Representations*, volume 2026, pp. 36144–36185, 2026b.

Yang Xiao, Mohan Jiang, Jie Sun, Keyu Li, Jifan Lin, Yumin Zhuang, Ji Zeng, Shijie Xia, Qishuo Hua, Xuefeng Li, et al. Limi: Less is more for agency. *arXiv preprint arXiv:2509.17567*, 2025.

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*, 2026.

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.

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

Shenzhi Yang, Guangcheng Zhu, Bowen Song, Haobo Wang, Mingxuan Xia, Xing Zheng, Yingfan Ma, Zhongqi Chen, Weiqiang Wang, Junbo Zhao, et al. Oprd: On-policy representation distillation. *arXiv preprint arXiv:2606.06021*, 2026b.

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*, 2026c.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. *Advances in Neural Information Processing Systems*, 35:20744–20757, 2022a.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022b.

Xinlei Yu, Gen Li, Qingyi Si, Guibin Zhang, Yuqi Xu, Congcong Wang, Shuai Dong, Kaiwen Tuo, Xiangyu Zeng, Kaituo Feng, et al. Dopd: Dual on-policy distillation. *arXiv preprint arXiv:2606.30626*, 2026.

Yanfei Zhang, Xu Lin, and Chenglin Wu. Stepopsd: Step-aware online preference distillation for agent reinforcement learning. *arXiv preprint arXiv:2605.27140*, 2026.

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-distilled reasoner: On-policy self-distillation for large language models. *arXiv preprint arXiv:2601.18734*, 2026.

AI USE STATEMENT

Large language models assist with editorial polishing of the manuscript. The authors determine the scientific claims, supply the experimental results, check the mathematical content, and remain responsible for the final text.

A ADDITIONAL DETAILS

A.1 INHERITED TRAJECTORY AND TOKEN-LEVEL OBJECTIVE

Local trajectory controller. Our implementation uses a local controller variant of trajectory-relayed on-policy distillation (Xu et al., 2026). During generation, a teacher span is triggered when the teacher argmax is a designated reflection token, differs from the sampled student token, and is absent from the student's top-5 predictions. Each response contains at most two teacher spans, and each teacher span is limited to three paragraphs and 256 tokens. These trajectories provide the response and teacher-token masks used by the stage construction in Section 3.3.

Base token-level objective. Let $h_{i,t}$ denote the trajectory prefix and $y_{i,t}$ the token emitted at position t. The current student and teacher define the token-level discrepancy and detached distillation advantage as

$$d_{i,t} = \log \pi_\theta(y_{i,t} \mid h_{i,t}) - \log \pi_T(y_{i,t} \mid h_{i,t}), \qquad A_{i,t} = -\operatorname{sg}(d_{i,t}), \tag{22}$$

where sg denotes stop-gradient. The importance ratio between the current student and the behavior policy is

$$\rho_{i,t} = \exp[\log \pi_\theta(y_{i,t} \mid h_{i,t}) - \log \pi_{\bar{\theta}}(y_{i,t} \mid h_{i,t})]\,. \tag{23}$$

The existing clipped policy-gradient objective (Schulman et al., 2017) maps $(\rho_{i,t}, A_{i,t})$ to the per-token loss $\ell_{i,t}$. STAGEOPD is applied only after $\ell_{i,t}$ has been computed, leaving the trajectory and token-level objective unchanged.

Implementation procedure. The implementation follows the stage-aware loss reweighting defined in Section 3.3. Given per-token loss ℓ, response mask R, teacher-token mask T, stage budgets, and Recovery window r, we: (i) construct the four mutually exclusive stage masks; (ii) divide each present stage budget uniformly among its tokens; (iii) normalize the resulting token weights to preserve unit mean weight within each response; and (iv) apply the normalized weights to ℓ before the final aggregation in Equation (13). The implementation additionally checks that the four masks are disjoint and jointly cover all valid response positions.

A.2 ADDITIONAL IMPLEMENTATION DETAILS

Table 8 lists implementation settings that complement the main experimental setup.

Table 8: **Additional implementation settings for STAGEOPD.**

Setting	Value
Global / PPO mini-batch	128 / 128
Maximum prompt / response length	2,048 / 16,384
STAGEOPD continuation learning rate	5×10^{-7}
Training seed	42
Trigger top-K	5
Maximum teacher interventions	2 per response
Maximum teacher-span length	3 paragraphs / 256 tokens

A.3 STAGE ALLOCATION ACROSS TRAINING

The main experiments report the aggregate allocation produced by STAGEOPD. Here we additionally examine representative training steps to show how the allocation behaves over time. Table 9 reports both token frequency and optimization-weight share for all four stages.

Table 9: **Stage allocation at representative training steps.** Token share measures stage frequency, while weight share measures the normalized optimization weight assigned by STAGEOPD.

Step	Statistic (%)	Normal	Trigger	Teacher Body	Recovery
66	Token share	97.531	0.0228	1.599	0.847
66	Weight share	78.854	0.587	14.685	5.874
70	Token share	97.677	0.0219	1.449	0.851
70	Weight share	81.124	0.524	13.108	5.243
75	Token share	97.661	0.0210	1.571	0.748
75	Weight share	81.704	0.510	12.757	5.028
80	Token share	98.206	0.0168	1.112	0.665
80	Weight share	80.971	0.529	13.215	5.286

Across steps 66–80, TEACHER BODY occupies only 1.112–1.599% of tokens while consistently receiving 12.757–14.685% of the optimization weight. RECOVERY similarly receives 5.028–5.874% of the weight while accounting for less than 0.9% of tokens. Meanwhile, NORMAL remains the dominant stage throughout training. These observations complement the aggregate allocation analysis in the main experiments and show that stage-aware reweighting remains effective across multiple training steps.

Earlier training also exhibits a pronounced decrease in teacher-token frequency as responses become longer. Table 10 reports a small set of representative points.

Table 10: **Teacher-token frequency during training.** Teacher interventions become increasingly sparse as average response length grows.

Step	Teacher tokens (%)	Response length
1	10.976	1,074.0
20	2.175	3,862.3
40	1.808	4,484.4
65	1.746	4,795.0

Teacher-token frequency decreases from 10.976% at the beginning of training to 1.746% at step 65, while the average response length grows substantially. This trend further motivates allocating optimization weight by structural stage rather than relying only on token frequency.

A.4 REPRESENTATIVE QUALITATIVE EXAMPLES

We provide two representative examples from benchmarks where STAGEOPD shows clear improvements over the fixed-token SARD-FULL baseline. They illustrate how the resulting model can maintain a useful reasoning direction through later generation.

Table 11: **Representative reasoning examples for STAGEOPD and the fixed-token SARD-FULL baseline.**

Problem	STAGEOPD	SARD-FULL
AMC 2023, Problem 7	$21/32$ **correct.** Recognizes that every composition of 10 has probability 2^{-10} and that there are 2^9 such compositions, yielding probability $1/2$ and answer 3.	$9/32$ correct. Uses a longer recurrence with repeated boundary-count errors and returns 97.
MATH500, Problem 486	$4/4$ **correct.** Identifies the sphere as inscribed in the minimum cube, giving side length 6 and volume $6^3 = 216$.	$0/4$ correct. Treats the sphere diameter as the cube's space diagonal and obtains $24\sqrt{3}$.

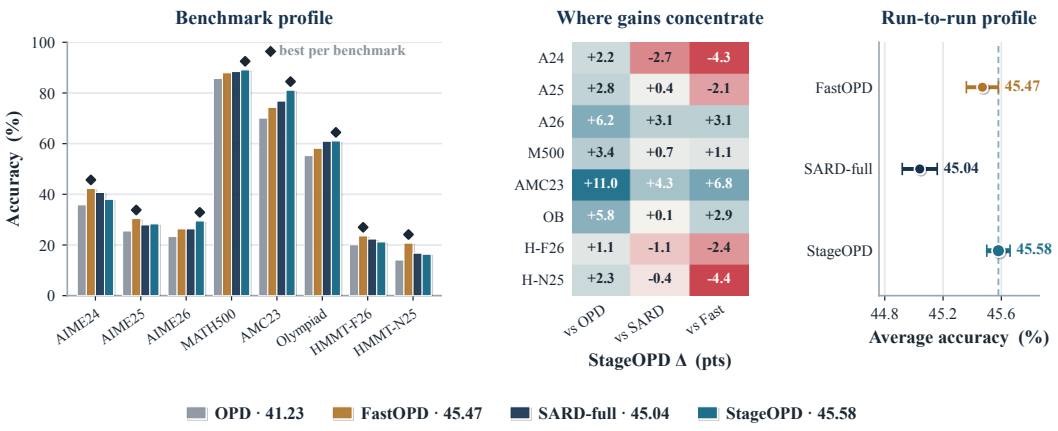


Figure 3: **Benchmark performance of StageOPD.** (a) Per-benchmark accuracy for OPD, FastOPD, SARD-full, and StageOPD. (b) Accuracy differences between StageOPD and representative OPD baselines, highlighting the strongest gains on AMC 2023, AIME 2026, MATH500, and OlympiadBench. (c) Mean and standard deviation of the eight-benchmark average across three runs, where StageOPD achieves the highest mean and lowest run-to-run variation.

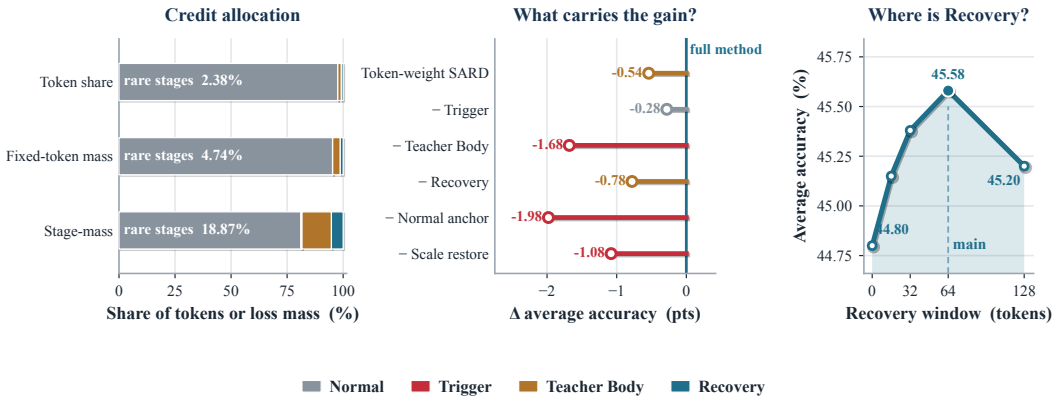


Figure 4: **Stage-aware allocation and component analysis.** (a) The three short stages occupy only about 2.38% of tokens but receive approximately 18.87% of the optimization weight under StageOPD. (b) Removing individual stage budgets or sequence-level normalization consistently reduces average accuracy. (c) A 64-token Recovery window achieves the best performance among the five tested settings.

The two examples complement the aggregate benchmark results by highlighting reasoning behaviors related to maintaining an appropriate intermediate direction after a local decision has been established.

A.5 Benchmark-level Improvements and Allocation Analysis

Figure 3 summarizes the benchmark-level improvements and the three-run results.

Figure 4 summarizes the allocation analysis, component ablations, and Recovery-window results.