arXiv:2610.00838v1 [cs.LG] 30 Sep 2026

SHARPO: Segment-Level Credit Assignment for Agentic Reinforcement Learning

Xinchen Du[1,2*] **Zhengze Zhou**[1] **Wenhui Zhu**[1]
Han Yu[1] **Sen Na**[2] **Rohit Jain**[1] **Alborz Geramifard**[1]
[1]LinkedIn Corporation [2]Georgia Institute of Technology

Abstract

Agentic reinforcement learning (RL) trains a large language model (LLM) to act over long, multi-step interactions. However, a single localized error can cause task failure, while trajectory-level rewards provide limited guidance for assigning credit to individual decisions. To address this limitation, we introduce Segment-level Hindsight Advantage Reweighting for Policy Optimization (SHARPO), a credit-assignment mechanism that refines Group Relative Policy Optimization (GRPO) at the level of environment-facing segments. Inspired by the existing on-policy self-distillation (OPSD) method, SHARPO computes teacher–student log-probability gaps within each segment and uses the resulting signal to compute a bounded multiplier on the GRPO advantage. This multiplier is shared by all tokens within the segment, allowing credit to vary across different segments. With Qwen2.5-7B-Instruct, SHARPO outperforms existing baselines on the ALFWorld and WebShop benchmarks, including GRPO, SDAR, RLSD, and StepOPSD.

1 Introduction

Post-training plays a key role in adapting large language models (LLMs) to specific tasks and improving their performance in interactive environments [Ouyang et al., 2022, Jin et al., 2025]. Reinforcement learning (RL) supports this adaptation by optimizing agent behavior using feedback from task outcomes. When these outcomes can be automatically verified, their correctness provides a direct reward signal for training, as demonstrated in mathematical reasoning and search-based question answering [Guo et al., 2025a, Jin et al., 2025]. Group Relative Policy Optimization (GRPO) offers an efficient approach to learning from such rewards without training a separate value network [Shao et al., 2024]. For each task, GRPO samples a group of rollouts and normalizes their rewards within the group to estimate advantages. Under outcome supervision, each rollout receives a single advantage that is shared by all of its generated tokens.

In multi-turn interactions, however, an agent must make a sequence of decisions before receiving feedback on the final outcome. In this case, the rewards are sparse and are available only at the end of an episode, hence they provide little direct information about the contribution of each intermediate step to that outcome. For example, a shopping agent may find a suitable product but select an incorrect option before completing the purchase. If the resulting trajectory receives a negative advantage, both the useful search and the mistaken selection inherit the same negative coefficient. The shared advantage therefore does not distinguish actions that helped accomplish the task from those that led to failure. This motivates us to design a credit-assignment mechanism in RL that uses finer-grained signals to refine the credit assigned to individual decisions within a trajectory.

One source of such fine-grained signals is on-policy distillation (OPD), where a student learns from a teacher's token-level predictions on responses generated by the student itself [Agarwal et al., 2024].

*Work done during an internship at LinkedIn Corporation.

Preprint.

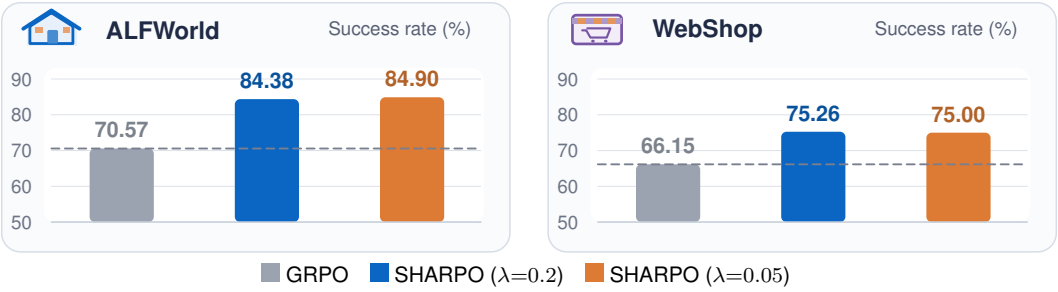

Figure 1: **Final validation success (%) with Qwen2.5-7B-Instruct.** The dashed line marks GRPO. SHARPO improves on GRPO at both mixing weights on both benchmarks, reaching 84.90% on ALFWorld ($+14.32$ points) and 75.26% on WebShop ($+9.11$ points). Bars are means over three independent runs at training step 200; the vertical axis starts at 50%. Table 1 reports all baselines with standard deviations.

This provides dense supervision along the student's own generation process, but typically relies on a separate, often larger, teacher model. On-policy self-distillation (OPSD) removes this dependence by using the same LLM as both teacher and student under different conditioning contexts [Zhao et al., 2026]. The teacher receives privileged information, such as a reference solution, and guides the student on its own rollouts. Therefore, OPSD provides dense training feedback without requiring an external teacher model.

Inspired by OPSD, our SHARPO reuses a successful rollout from the same GRPO group to show the teacher how the task can be completed. The teacher uses this additional information to evaluate the student's behavior and provide feedback for credit assignment within the trajectory. Concretely, we instantiate the teacher as a copy of the agent's policy and compute teacher–student log-probability differences. These differences indicate how strongly the teacher supports the sampled behavior relative to the student. We then use this signal to adjust the magnitude of GRPO's outcome-derived advantages. To sum up, our method SHARPO combines fine-grained guidance from self-distillation with task-level supervision from environmental rewards.

In an interactive task, a trajectory contains semantically meaningful spans, such as agent reasoning, tool calls, and actions. We refer to these spans as interaction segments and use those as the units for refining credit assignment. For each selected segment, SHARPO assigns a weight based on the teacher–student log-probability difference for that segment. This weight scales the GRPO advantage uniformly within its segment, refining how the trajectory-level reward outcome is credited to intermediate substeps. By incorporating the teacher's assessment of each substep, SHARPO provides local guidance on the strength of reinforcement or penalization, alleviating the credit-assignment problem caused by sparse terminal rewards.

This segment-level formulation distinguishes SHARPO from related approaches in how teacher feedback is organized and used for credit assignment. Unlike RLSD [Yang et al., 2026], which reweights token-level advantages using self-divergence, or SDAR [Lu et al., 2026], which applies soft token-level gating to an auxiliary distillation loss over the full response, SHARPO aggregates teacher feedback within selected interaction segments and assigns a shared advantage weight to each segment. Compared to StepOPSD [Zhang et al., 2026], which retains token-specific weights within selected action-centered spans, SHARPO aggregates the feedback within each segment before determining its weight. Favorable and unfavorable evidence across a substep is therefore considered jointly, allowing the credit adjustment to reflect the teacher's overall support for the complete decision.

We evaluate SHARPO on ALFWorld and WebShop using Qwen2.5-7B-Instruct. As shown in Figure 1, SHARPO achieves higher final validation success rates than baselines on both benchmarks at both parameter configurations. To summarize, our main contributions are threefold:

- We propose SHARPO, a segment-level credit-assignment method that uses successful peer rollouts to guide the weighting of intermediate decisions in GRPO. By aggregating self-distillation feedback over meaningful interaction segments, it supplements sparse outcome supervision without requiring an external teacher or additional environment rollouts.

- We provide a theoretical explanation of why this self-distillation feedback is useful in our method and how it supports finer-grained credit assignment.
- We demonstrate that SHARPO outperforms GRPO, RLSD, SDAR, and StepOPSD on both ALFWorld and WebShop. Analyses of subtask success rates and product-matching quality further characterize these gains.

2 Method

We first review GRPO and OPSD in Section 2.1 and then present SHARPO in Section 2.2. Section 2.3 provides a theoretical analysis of our segment-level credit assignment.

2.1 Preliminaries: GRPO and OPSD

We consider an LLM agent initialized from a pretrained model, with trainable parameters θ. Its autoregressive policy π_θ defines a probability distribution over the next token given the available context. Starting from a task prompt x, the agent generates responses and receives observations from the environment after executing actions. Together, these responses and observations form an interaction trajectory τ. Let y_ℓ denote the ℓ-th token generated by the agent across τ, and let T be the total number of such tokens. Each token is sampled as $y_\ell \sim \pi_\theta(\cdot \mid c_\ell)$, where c_ℓ contains the prompt x, previously generated tokens, and environment observations available before y_ℓ. At the end of the episode, a verifier assigns $r(\tau) = 1$ if the task succeeds and 0 otherwise. We use RL post-training methods to update the policy parameters θ so as to maximize the expected terminal reward.

Group Relative Policy Optimization (GRPO). GRPO [Shao et al., 2024] samples G trajectories $\{\tau_i\}_{i=1}^{G}$ for the same task using the behavior policy $\pi_{\theta_{\text{old}}}$, where θ_{old} denotes the parameters saved before sampling and held fixed during subsequent updates. For rewards $r_i = r(\tau_i)$, it computes the trajectory advantage

$$A_i = \frac{r_i - \text{mean}(r_1, \ldots, r_G)}{\text{std}(r_1, \ldots, r_G) + \epsilon}. \tag{1}$$

Here $\epsilon > 0$ is a numerical stabilizer, and the advantage A_i measures whether a trajectory performs better or worse than its group average and is shared by all T_i generated tokens in that trajectory. We write $\omega_{i,\ell}(\theta) = \pi_\theta(y_{i,\ell} \mid c_{i,\ell})/\pi_{\theta_{\text{old}}}(y_{i,\ell} \mid c_{i,\ell})$ for the probability ratio between the current and behavior policies. Hence we have

$$\ell_{i,\ell}(\theta; A) = \min\big\{\omega_{i,\ell}(\theta)A,\ \text{clip}\big(\omega_{i,\ell}(\theta), 1-\varepsilon, 1+\varepsilon\big)A\big\}, \tag{2}$$

where the clip function truncates the ratio to $[1-\varepsilon, 1+\varepsilon]$. Finally, GRPO maximizes

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}\left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{T_i}\sum_{\ell=1}^{T_i}\ell_{i,\ell}(\theta; A_i)\right] - \beta_{\text{KL}}\mathcal{K}(\theta), \tag{3}$$

where the expectation is over tasks and sampled trajectories, and ε is the ratio-clipping radius. The term $\mathcal{K}(\theta)$ denotes the sampled KL penalty against a fixed reference policy π_{ref}, with its strength controlled by $\beta_{\text{KL}} \geq 0$. Group normalization provides the baseline without requiring a learned value function.

On-Policy Self-Distillation (OPSD). OPSD [Zhao et al., 2026] uses the same LLM as student and teacher under different conditioning contexts. For a prompt x with reference information h, the student samples a response y from $\pi_\theta(\cdot \mid x)$, while the teacher additionally receives h. A basic distillation objective is

$$\mathcal{L}_{\text{OPSD}}(\theta) = \mathbb{E}_{x,h,y}\left[\frac{1}{|y|}\sum_{\ell=1}^{|y|}D(\pi_\theta(\cdot \mid c_\ell \oplus h) \,\|\, \pi_\theta(\cdot \mid c_\ell))\right], \tag{4}$$

where $c_\ell = (x, y_{<\ell})$, $c_\ell \oplus h = (x \oplus h, y_{<\ell})$ augments the prompt with reference information while retaining the sampled prefix, and D is a distributional divergence. The sampled prefixes and teacher distributions are held fixed when differentiating this objective, hence gradients flow only through

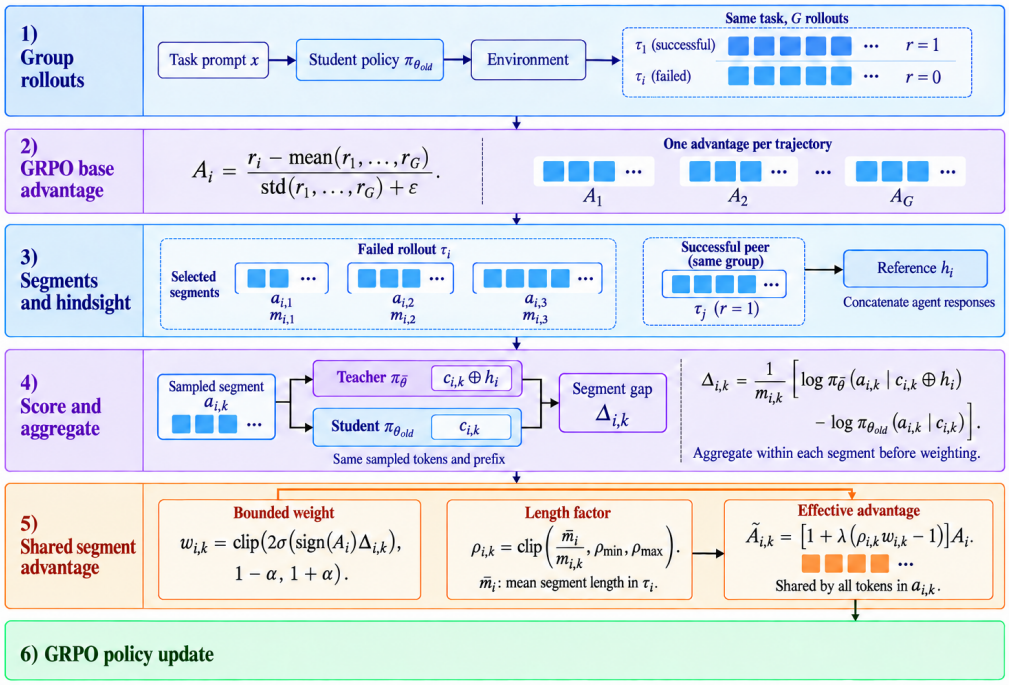

Figure 2: **SHARPO Framework.** A successful peer supplies context for a policy-copy teacher. A length-normalized likelihood comparison of each selected segment produces a bounded weight. Length balancing and interpolation with the base advantage then determine the segment's effective advantage for the policy update.

the student predictions. Evaluating student-generated prefixes supplies local supervision without an external teacher model.

With terminal rewards, GRPO assigns the same trajectory advantage in Eq. (1) to every generated token. This shared signal cannot distinguish useful intermediate decisions from local mistakes within the same rollout. OPSD provides an additional source of supervision by conditioning the teacher on a reference solution. The teacher can provide different guidance at different positions in the student's response, supplying local information that the terminal reward alone does not provide. Inspired by this idea, SHARPO uses successful peer rollouts to obtain teacher feedback and aggregates it within interaction segments. This feedback adjusts how strongly each segment is reinforced or penalized, offering an approach to addressing the credit-assignment problem under sparse terminal rewards.

2.2 SHARPO

After rollout collection and reward evaluation, SHARPO uses teacher feedback to reweight the GRPO base advantage at the segment level. The adjusted advantages are then used in the objective in Eq. (3), retaining its clipped surrogate and KL regularization. Figure 2 illustrates the overall training procedure.

Step 1: identify interaction segments. An interaction segment is a contiguous span of the agent's response expressing a coherent unit of behavior, such as an action enclosed in `<action>...</action>` or a reasoning passage enclosed in `<thinking>...</thinking>`. For the i-th trajectory τ_i, we select K_i nonempty, disjoint segments for credit assignment. Let $a_{i,k}$ denote the k-th selected segment and $m_{i,k} = |a_{i,k}|$ its length. We denote by $c_{i,k}$ the context available to the agent immediately before generating $a_{i,k}$. It consists of the task prompt x and all preceding agent-generated content and environment observations, including any content earlier in the same response. We assess and assign credit to each selected segment as a whole.

Step 2: construct a teacher context from successful peers. Within each group, we select one successful rollout and construct a reference text by concatenating its agent responses in interaction order, separated by step labels. For example, a successful shopping rollout may provide responses covering product search, option selection, and purchase. All lower-reward trajectories in the group share this reference text. We denote the reference used to evaluate trajectory τ_i by h_i. The teacher $\pi_{\bar{\theta}}$ is a copy of the trainable student policy π_θ. Its parameters $\bar{\theta}$ are copied from the student at each teacher refresh and held fixed until the next refresh.

Step 3: assess each segment. The teacher assesses the student's sampled behavior with additional reference information. We write $c_{i,k} \oplus h_i$ for the teacher's context, which augments the original interaction context $c_{i,k}$ with the successful reference h_i. The student receives only $c_{i,k}$. Their teacher–student log-probability gap is

$$\Delta_{i,k} = \frac{1}{m_{i,k}} \left[\log \pi_{\bar{\theta}}(a_{i,k} \mid c_{i,k} \oplus h_i) - \log \pi_\theta(a_{i,k} \mid c_{i,k})\right]. \tag{5}$$

The student likelihood is evaluated at the rollout parameters $\theta = \theta_{\text{old}}$, and the gap is held fixed during the subsequent policy update. A positive gap indicates stronger support for the segment from the reference-informed teacher than from the student, while a negative gap indicates weaker support. Such feedback can vary across segments within a failed trajectory, even though they share the same outcome-derived advantage. We therefore use the gap to determine each segment's weight in Step 4, adjusting its contribution to the parameter update according to the teacher's local feedback.

Step 4: compute segment weights. With the sigmoid function $\sigma(z) = (1 + e^{-z})^{-1}$, we map the teacher–student gap to a bounded weight:

$$w_{i,k} = \text{clip}(2\sigma(\text{sign}(A_i)\Delta_{i,k}), 1-\alpha, 1+\alpha), \qquad 0 < \alpha < 1. \tag{6}$$

The sigmoid function smoothly converts the signed gap into a positive weight, and the factor of 2 ensures that zero gap $\Delta_{i,k} = 0$ gives a weight of 1. To balance the shaped coefficients across segment lengths, we use

$$\rho_{i,k} = \text{clip}\left(\frac{\bar{m}_i}{m_{i,k}}, \rho_{\min}, \rho_{\max}\right), \qquad \bar{m}_i = \frac{1}{K_i}\sum_{k=1}^{K_i} m_{i,k}, \tag{7}$$

where $0 < \rho_{\min} \leq \rho_{\max}$. This factor compensates for differences in segment length, reducing the influence of length on the total credit assigned to each segment.

Step 5: update the policy. We interpolate between the base and reweighted advantages:

$$g_{i,k} = 1 + \lambda(\rho_{i,k} w_{i,k} - 1), \qquad \widetilde{A}_{i,k} = g_{i,k} A_i, \qquad 0 \leq \lambda \leq 1. \tag{8}$$

We then update the student parameters θ by maximizing the objective in Eq. (3) with these segment-level advantages $\widetilde{A}_{i,k}$.

2.3 Understanding Segment-Level Credit Assignment

We use an ideal success-conditioned reference to explain how hindsight feedback refines credit for individual decisions. Let $u(z) = \text{clip}(2\sigma(z) - 1, -\alpha, \alpha)$, a bounded transform that preserves the sign of its input. Its odd symmetry gives

$$\widetilde{A}_{i,k} = \underbrace{A_i}_{\text{outcome}} + \underbrace{\lambda A_i(\rho_{i,k} - 1)}_{\text{length adjustment}} + \underbrace{\lambda \rho_{i,k} |A_i| u(\Delta_{i,k})}_{\text{hindsight adjustment}}. \tag{9}$$

For a fixed segment, we suppress (i, k) and write $B = (1 - \lambda + \lambda\rho)A$ for the coefficient without hindsight. The remaining term, $\lambda\rho|A|u(\Delta)$, raises or lowers this coefficient according to the segment's feedback.

To judge this adjustment, let $\pi = \pi_{\theta_{\text{old}}}$ and consider a segment a of length $m > 0$ at context c. Define $Q^\pi(c, a) = \Pr_\pi(R = 1 \mid c, a)$ as the success probability after choosing a and then following π, where $R = 1$ denotes terminal success. The baseline $V^\pi(c) = \Pr_\pi(R = 1 \mid c)$ is the success probability conditioned on previous context c. Their difference, $D^\pi(c, a) = Q^\pi(c, a) - V^\pi(c)$, is the

segment's local value. Thus $D^\pi > 0$ means choosing a improves success probability, and $D^\pi < 0$ means it reduces it.

We now consider the ideal reference $q^\star(a \mid c) = \Pr_\pi(a \mid c, R = 1)$. Successful-peer conditioning aims to approximate this assessment for the failed rollouts selected in Step 2. We now use Proposition 1 to explain why the hindsight information is useful for refined credit assignment.

Proposition 1. *Assume $V^\pi(c) > 0$ and $\pi(a \mid c) > 0$. Let $\Delta^\star = m^{-1}[\log q^\star(a \mid c) - \log \pi(a \mid c)]$ and $\widetilde{A}^\star = B + \lambda\rho|A|u(\Delta^\star)$.*

(a) The reference gap satisfies

$$\frac{q^\star(a \mid c)}{\pi(a \mid c)} = \frac{Q^\pi(c,a)}{V^\pi(c)}, \qquad \operatorname{sign} u(\Delta^\star) = \operatorname{sign} D^\pi(c,a). \tag{10}$$

(b) For any $\rho > 0$, the refined coefficient satisfies

$$[-\widetilde{A}^\star D^\pi(c,a)]_+ \le [-BD^\pi(c,a)]_+, \tag{11}$$

where $[z]_+ = \max\{z, 0\}$.

In Part (a), Equation (10) shows that, under the ideal reference, $u(\Delta^\star)$ and the segment's local value $D^\pi(c,a)$ have the same direction. Therefore, scaling this signal $u(\Delta^\star)$ by $\lambda\rho|A|$, the hindsight adjustment (cf. Eq. (9)) also aligns with the direction of the local value $D^\pi(c,a)$. This means a positive hindsight adjustment identifies a segment that increases the final success probability, while a negative hindsight adjustment identifies one that decreases it. This alignment explains why we use the teacher–student gap as hindsight feedback. After length normalization and the sign-preserving transform u, the ideal signal $\Delta^\star$ indicates whether each segment helps or harms task success. This local information helps refine credit assignment within the trajectory.

In Part (b), B stands for the segment's advantage adjustment without hindsight feedback. When we choose the length normalization term $\rho = 1$, B becomes the base GRPO advantage A. The quantity $[-BD^\pi(c,a)]_+$ measures the conflict between this coefficient B and the segment's local value D^π. It is positive when B opposes D^π and zero otherwise. When a useful decision in a failed rollout has $D^\pi(c,a) > 0$ but receives the advantage $B < 0$, we have $[-BD^\pi(c,a)]_+ > 0$. This indicates a credit-assignment error, as a useful decision receives negative credit because its trajectory fails. Equation (11) shows that ideal hindsight does not worsen this error for any $\rho > 0$.

3 Experiments

We conduct experiments on ALFWorld and WebShop to demonstrate the effectiveness of SHARPO through comparisons with existing RL baselines.

3.1 Setup

We fine-tune Qwen2.5-7B-Instruct [Yang et al., 2024] on two interactive benchmarks, ALFWorld [Shridhar et al., 2021] and WebShop [Yao et al., 2022]. ALFWorld covers six families of household tasks. WebShop requires agents to search for and purchase products that satisfy user requests. Both environments use binary terminal success rewards for training.

All methods share the same backbone, prompts, environments, reward function, optimizer, rollout settings, and evaluation protocol. We train each configuration for 200 optimization steps with 16 tasks per batch, a learning rate of 5×10^{-7}, and a KL coefficient of $\beta_{\mathrm{KL}} = 0.01$. For SHARPO, we set $\alpha = 0.2$, $\rho_{\min} = 0.25$, and $\rho_{\max} = 4$. We evaluate two mixing weights, $\lambda \in \{0.2, 0.05\}$, each held fixed throughout training.

Our primary metric is the final validation success rate at step 200, evaluated on 128 held-out tasks. We report the mean and sample standard deviation over three independent runs. To examine performance in more detail, we also report subtask success rates by ALFWorld task family and the continuous task score in WebShop, which measures how well the purchased product matches the request.

We compare our SHARPO with GRPO [Shao et al., 2024], which assigns a single outcome-derived advantage to all generated tokens within each trajectory. We also consider baselines that combine RL

Method	ALFWorld		WebShop	
	Success rate (%)	Gain	Success rate (%)	Gain
GRPO	70.57 ± 7.83	–	66.15 ± 1.19	–
SDAR	67.19 ± 9.50	-3.39	61.72 ± 6.10	-4.43
RLSD	73.96 ± 5.76	$+3.39$	69.01 ± 6.31	$+2.86$
GRPO+OPSD	72.92 ± 4.51	$+2.34$	69.01 ± 3.16	$+2.86$
StepOPSD ($\lambda = 0.2$)	77.60 ± 2.96	$+7.03$	69.27 ± 1.63	$+3.12$
StepOPSD ($\lambda = 0.05$)	78.12 ± 3.41	$+7.55$	67.45 ± 4.01	$+1.30$
SHARPO ($\lambda = 0.2$)	84.38 ± 2.82	$+13.80$	75.26 ± 1.19	$+9.11$
SHARPO ($\lambda = 0.05$)	84.90 ± 1.19	$+14.32$	75.00 ± 1.35	$+8.85$

Table 1: **Final validation success rate (%)** at step 200 with Qwen2.5-7B-Instruct, mean $\pm$ sample standard deviation over three independent runs. Gain is the change in mean success rate relative to GRPO, in percentage points. Red and blue indicate the best and second-best mean results in each column, respectively.

with self-distillation: RLSD [Yang et al., 2026] uses self-distillation gaps to reweight advantages at the token level, while SDAR [Lu et al., 2026] applies token-level gating to an auxiliary distillation loss. GRPO+OPSD augments the GRPO objective with the OPSD distillation loss [Zhao et al., 2026]. Finally, StepOPSD [Zhang et al., 2026] uses hindsight feedback to compute token-specific advantage weights within selected interaction spans.

3.2 Main results

Table 1 reports the overall comparison. SHARPO achieves the highest mean final validation success rate on both benchmarks at both tested mixing weights. Relative to GRPO, the gains are 13.80–14.32 percentage points on ALFWorld and 8.85–9.11 points on WebShop. Compared with StepOPSD at the corresponding reported λ values, SHARPO gains 6.77 points on ALFWorld at both settings and 5.99–7.55 points on WebShop. These results support the effectiveness of using segment-level self-distillation feedback to refine GRPO credit assignment.

The improvements persist across both tested mixing weights. Although $\lambda = 0.2$ and $\lambda = 0.05$ differ by a factor of four, their mean success rates differ by only 0.52 percentage points on ALFWorld and 0.26 points on WebShop. Across the three independent runs, SHARPO has lower variability than GRPO on ALFWorld, with standard deviations of 2.82 and 1.19 points compared with 7.83. On WebShop, its standard deviations of 1.19 and 1.35 points are comparable to GRPO's 1.19. The higher mean success rates therefore accompany lower run-to-run variability on ALFWorld and similar variability on WebShop.

3.3 Fine-grained analysis

Subtask performance. Table 2 shows that SHARPO performs strongly across the six ALFWorld task families. At both tested mixing weights $\lambda = 0.2$ and $\lambda = 0.05$, it outperforms all evaluated baselines on Pick, Clean, Cool, and Pick2, and remains within 0.30 percentage points of the best baseline on Look. These broad gains persist across the two mixing weights, which differ by a factor of four.

The improvements extend to complicated tasks such as Heat, Cool, and Pick2, which requires object-state changes or the handling of multiple objects. On Pick2, SHARPO exceeds the strongest baseline by 7.02 and 10.10 percentage points at $\lambda = 0.2$ and 0.05, respectively. On Cool, $\lambda = 0.2$ improves over the strongest baseline by 6.89 percentage points. On Heat, $\lambda = 0.05$ exceeds SDAR, the strongest baseline for this task, by 6.67 percentage points. These tasks require several coordinated decisions, and a final failure can follow both useful intermediate actions and local mistakes. The gains are consistent with the motivation for segment-level credit assignment, which uses successful-peer feedback to differentiate the training weights of intermediate decisions.

Method	Pick	Look	Clean	Heat	Cool	Pick2	All
GRPO	89.22	73.33	81.39	61.90	58.21	60.06	70.57
SDAR	80.94	82.22	76.55	71.43	54.68	48.00	67.19
RLSD	89.65	74.44	88.49	55.24	55.04	70.71	73.96
GRPO+OPSD	90.95	81.72	77.92	39.05	62.29	66.04	72.92
StepOPSD ($\lambda = 0.2$)	90.96	83.33	83.50	52.86	68.46	75.38	77.60
StepOPSD ($\lambda = 0.05$)	90.32	85.86	88.60	69.52	65.70	64.67	78.12
SHARPO ($\lambda = 0.2$)	93.73	85.56	91.49	61.67	75.35	82.40	84.38
SHARPO ($\lambda = 0.05$)	96.00	85.56	92.68	78.10	68.82	85.48	84.90

Table 2: **ALFWorld success rate (%) by task family** at step 200, mean over three independent runs. Family rates are averaged separately; *All* reports the mean overall validation success rate from Table 1. Red and blue indicate the best and second-best mean results in each column, respectively.

Product-matching quality in WebShop. Success rate measures whether a purchase fully satisfies the shopping request. WebShop's task score provides a finer-grained assessment of how well the purchase meets that request, giving credit to partial matches as well as complete success [Yao et al., 2022, Section 3.1]. The score of one evaluation episode is

$$S = r_{\text{type}} \frac{|U_{\text{att}} \cap Y_{\text{att}}| + |U_{\text{opt}} \cap Y_{\text{opt}}| + \mathbf{1}\{y_{\text{price}} \leq u_{\text{price}}\}}{|U_{\text{att}}| + |U_{\text{opt}}| + 1}. \tag{12}$$

Here U_{att} and U_{opt} are the requested attributes and options respectively. Y_{att} and Y_{opt} describe the purchased product and selected options respectively. The intersections count matches under the benchmark's rules. The product price y_{price} is compared with the budget u_{price}, and $r_{\text{type}} \in [0, 1]$ measures product-type matching. Episodes without a purchase receive zero. At training step 200, we average S over all 128 validation episodes to obtain each run's task score. We then report the mean of these scores on a $[0, 1]$ scale. Task score captures how well a purchase satisfies the request, even when success rate counts it as a failure. For example, a shirt may match the requested material, color, and budget but have the wrong size. A second shirt may meet the same budget but have the wrong size, material, and color. Both purchases count as failures, but task score credits the additional requirements satisfied by the first.

Table 3 shows gains over StepOPSD of 0.0591 and 0.0212 at $\lambda = 0.2$ and 0.05, respectively. Both settings also exceed GRPO. Thus, higher success rates are accompanied by better average satisfaction of the shopping requirements. Unlike StepOPSD's token-specific weights, our method assigns a shared hindsight-derived weight to each complete interaction segment. The higher task scores are consistent with evaluating search, option-selection, and purchase decisions at this shared granularity.

Method	Task score
GRPO	0.7781
StepOPSD ($\lambda = 0.2$)	0.8231
StepOPSD ($\lambda = 0.05$)	0.8268
SHARPO ($\lambda = 0.2$)	0.8822
SHARPO ($\lambda = 0.05$)	0.8480

Table 3: **WebShop task score.** Higher is better. Red and blue mark the best and second-best results among the listed methods.

4 Related Work

Outcome-supervised RL has been applied to reasoning models [Shao et al., 2024, Guo et al., 2025a] and multi-turn language agents [Jin et al., 2025, Wang et al., 2025]. Actor-critic implementations of PPO [Schulman et al., 2017] commonly pair a learned value function with generalized advantage estimation (GAE) [Schulman et al., 2016], which combines temporal-difference residuals across

time in a construction related to eligibility traces. ArCHer treats utterances as high-level actions and uses off-policy critic learning to obtain utterance-level advantages for token-level policy updates [Zhou et al., 2024]. Other approaches use recurring environment states [Feng et al., 2025], Monte Carlo resampling [Kazemnejad et al., 2025, Guo et al., 2025b], or process reward models [Lightman et al., 2024, Wang et al., 2024]. TRIAGE assigns credit to role-typed environment-facing segments [Xu et al., 2026], while POAD studies action-level optimization [Wen et al., 2024]. In comparison, our method SHARPO retains GRPO's critic-free trajectory advantage and obtains local feedback by scoring sampled segments under student and successful-peer-conditioned teacher policies. The resulting log-probability gaps determine bounded weights shared within each selected segment. This provides segment-specific credit without fitting a critic to expected task returns or using temporal-difference backups, external judges, or additional environment rollouts.

This use of successful-peer context is related to context distillation, which transfers behavior induced by privileged context [Snell et al., 2022]. On-policy distillation evaluates the student's own samples [Agarwal et al., 2024]. OPSD uses the same model under different conditioning contexts [Zhao et al., 2026]. RLSD applies self-distillation gaps to token-level advantage magnitudes [Yang et al., 2026]; SDAR gates an auxiliary distillation objective [Lu et al., 2026]; and StepOPSD uses successful-peer hindsight for token-specific weighting within action-centered spans [Zhang et al., 2026]. SHARPO instead aggregates feedback over each selected segment before weighting its GRPO advantage. Concurrent AgentOPSD aggregates feedback over full turns using a retrieved skill and a recursive belief update [Wang et al., 2026]; our method uses independently scored environment-facing segments and successful peers from the current group.

5 Conclusion

SHARPO refines GRPO credit using successful-peer feedback on intermediate interaction segments. It aggregates teacher–student log-probability gaps within each segment and applies a shared, bounded adjustment to the outcome-derived advantage. The analysis explains how ideal success-conditioned feedback refines local credit. On ALFWorld and WebShop, the method improves final validation success over GRPO and the evaluated self-distillation baselines at both tested mixing weights. These results support using meaningful interaction segments to organize self-distillation feedback for agent training.

References

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

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.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Hanwei Xu, Honghui Ding, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jingchang Chen, Jingyang Yuan, Jinhao Tu, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaichao You, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingxu Zhou, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao,

Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645(8081):633–638, September 2025a. ISSN 1476-4687. doi: 10.1038/s41586-025-09422-z.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

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*, 2024.

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. In *Forty-third International Conference on Machine Learning*, 2026.

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*, 2026.

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

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

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning text and embodied environments for interactive learning. In *International Conference on Learning Representations*, 2021.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. WebShop: Towards scalable real-world web interaction with grounded language agents. In *Advances in Neural Information Processing Systems*, 2022.

Zihan Wang, Kangrui Wang, Qineng Wang, Pingyue Zhang, Linjie Li, Zhengyuan Yang, Xing Jin, Kefan Yu, Minh Nhat Nguyen, Licheng Liu, Eli Gottlieb, Yiping Lu, Kyunghyun Cho, Jiajun Wu, Li Fei-Fei, Lijuan Wang, Yejin Choi, and Manling Li. RAGEN: Understanding self-evolution in LLM agents via multi-turn reinforcement learning. *arXiv preprint arXiv:2504.20073*, 2025.

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

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. In *International Conference on Learning Representations (ICLR)*, 2016.

Yifei Zhou, Andrea Zanette, Jiayi Pan, Sergey Levine, and Aviral Kumar. ArCHer: Training language model agents via hierarchical multi-turn RL. In *International Conference on Machine Learning (ICML)*, 2024.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for LLM agent training. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Amirhossein Kazemnejad, Milad Aghajohari, Eva Portelance, Alessandro Sordoni, Siva Reddy, Aaron Courville, and Nicolas Le Roux. VinePPO: Refining credit assignment in RL training of LLMs. In *International Conference on Machine Learning (ICML)*, 2025.

Yiran Guo, Lijie Xu, Jie Liu, Dan Ye, and Shuang Qiu. Segment policy optimization: Effective segment-level credit assignment in RL for large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025b.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri 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*, 2024.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Y. Wu, and Zhifang Sui. Math-Shepherd: Verify and reinforce LLMs step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024.

Yuanda Xu, Zhengze Zhou, Hejian Sang, Xiaomin Li, Jiaxin Zhang, Xinchen Du, Sen Na, Zhipeng Wang, and Alborz Geramifard. TRIAGE: Role-typed credit assignment for agentic reinforcement learning. *arXiv preprint arXiv:2606.32017*, 2026. URL `https://arxiv.org/abs/2606.32017`.

Muning Wen, Ziyu Wan, Jun Wang, Weinan Zhang, and Ying Wen. Reinforcing LLM agents via policy optimization with action decomposition. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Charlie Snell, Dan Klein, and Ruiqi Zhong. Learning by distilling context. *arXiv preprint arXiv:2209.15189*, 2022.

Zi-Han Wang, Zhengxi Lu, Zhiyuan Yao, Jinyang Wu, Jie Wu, Zhengzhou Cai, Yueqing Sun, Ziang Ye, Linji Hao, Qi Gu, Xunliang Cai, Yongliang Shen, and Yujiu Yang. AgentOPSD: Recursive self-distillation for agentic reinforcement learning. *arXiv preprint arXiv:2608.05987*, 2026.

A Theory of segment-level credit refinement

We use the notation of Section 2.3. Sampled trajectories, base advantages, teacher feedback, and segment weights are held fixed during each policy update. Unless indices are needed, we analyze one selected segment of length $m > 0$, with $\pi = \pi_{\theta_{\text{old}}}$.

A.1 Value-aligned credit correction

Proof of Proposition 1. The function $u(z) = \text{clip}(2\sigma(z) - 1, -\alpha, \alpha)$ is odd, preserves sign, and satisfies $|u(z)| \leq \alpha < 1$. Thus

$$w - 1 = u(\text{sign}(A)\Delta), \qquad A(w-1) = |A|u(\Delta).$$

Substituting these identities into Eq. (8) gives

$$\widetilde{A} = (1 - \lambda + \lambda\rho)A + \lambda\rho|A|u(\Delta),$$

which proves Eq. (9).

For the success-conditioned reference, Bayes' rule gives

$$q^\star(a \mid c) = \frac{\Pr_\pi(a, R = 1 \mid c)}{\Pr_\pi(R = 1 \mid c)} = \pi(a \mid c)\frac{Q^\pi(c, a)}{V^\pi(c)}.$$

When $Q^\pi(c,a) > 0$, monotonicity of the logarithm implies

$$\operatorname{sign}\Delta^\star = \operatorname{sign}\left(\log \frac{Q^\pi(c,a)}{V^\pi(c)}\right) = \operatorname{sign} D^\pi(c,a).$$

Since u preserves sign, Eq. (10) follows. When $Q^\pi(c,a) = 0$, the same conclusion holds with $u(\Delta^\star) = -\alpha$. The correction $\lambda\rho|A|u(\Delta^\star)$ consequently has the direction of $D^\pi(c,a)$ whenever it is nonzero.

For part (b), write $D = D^\pi(c,a)$. Part (a) gives $u(\Delta^\star)D \geq 0$. For any $\rho > 0$ and $0 \leq \lambda \leq 1$,

$$(\widetilde{A}^\star - B)D = \lambda\rho|A|u(\Delta^\star)D \geq 0.$$

Thus $-\widetilde{A}^\star D \leq -BD$. Since $z \mapsto [z]_+$ is nondecreasing,

$$[-\widetilde{A}^\star D]_+ \leq [-BD]_+,$$

which proves Eq. (11), including the cases $A = 0$, $D = 0$, or $\lambda = 0$.

To see how this correction acts within a failed rollout, take $A < 0$ and $\lambda > 0$. Then $B < 0$, and odd symmetry gives

$$\widetilde{A}^\star = A\big[1 - \lambda + \lambda\rho(1 - u(\Delta^\star))\big] < 0,$$

since $1 - u(\Delta^\star) \geq 1 - \alpha > 0$. If $D > 0$, part (a) gives a positive correction, so $B < \widetilde{A}^\star < 0$. If $D < 0$, the correction is negative, so $\widetilde{A}^\star < B < 0$. These relations establish the weaker and stronger penalties described in the main text at the same length factor. □

A.2 Computing segment feedback

Let $a_{i,k} = (z_1, \ldots, z_m)$ be a sampled segment with $m = m_{i,k} > 0$. Assume both policies assign positive likelihood to it under their respective contexts. The teacher and student use the same sampled prefix $z_{<j}$, with the student evaluated at θ_{old} and the teacher additionally conditioned on h_i. By autoregressive factorization, the length-normalized segment gap in Eq. (5) equals the average of its conditional token gaps:

$$\begin{aligned} \Delta_{i,k} &= \frac{1}{m}\sum_{j=1}^{m} \delta_j, \\ \delta_j &= \log \pi_{\bar\theta}(z_j \mid c_{i,k} \oplus h_i, z_{<j}) - \log \pi_{\theta_{\text{old}}}(z_j \mid c_{i,k}, z_{<j}). \end{aligned} \tag{13}$$

A.3 Length-balancing rationale

We compare coefficient totals over $K_i > 0$ nonempty selected segments within one trajectory, omitting their common normalization $1/T_i$. Equation (8) gives $\widetilde{A}_{i,k} = (1-\lambda)A_i + \lambda\rho_{i,k}w_{i,k}A_i$, with $0 \leq \lambda \leq 1$. Without length balancing, the reweighted component accumulates absolute coefficient $\lambda m_{i,k}w_{i,k}|A_i|$, which contains an explicit factor of segment length. Suppose all length factors are unclipped, so $\rho_{i,k} = \bar m_i / m_{i,k}$ with $\bar m_i = K_i^{-1}\sum_{k=1}^{K_i} m_{i,k}$. Then

$$\lambda m_{i,k}\rho_{i,k}w_{i,k}|A_i| = \lambda \bar m_i w_{i,k}|A_i|. \tag{14}$$

Thus, equally weighted segments in the same trajectory receive the same accumulated coefficient in the reweighted component, regardless of length. The base component $(1-\lambda)A_i$ remains unchanged. This balances coefficients in the reweighted component, not the full update or segment gradient norms.

The numerator $\bar m_i$ preserves the overall coefficient scale. With $\bar w_i = K_i^{-1}\sum_{k=1}^{K_i} w_{i,k}$, substitution into Eq. (8) and $\sum_k m_{i,k} = K_i\bar m_i$ give

$$\sum_{k=1}^{K_i} m_{i,k}\widetilde{A}_{i,k} = \big[1 + \lambda(\bar w_i - 1)\big]\sum_{k=1}^{K_i} m_{i,k}A_i. \tag{15}$$

When all $w_{i,k} = 1$, the total advantage coefficient over selected segments is unchanged. Weight clipping in Eq. (6) remains in effect, so $w_{i,k} \in [1-\alpha, 1+\alpha]$ and the common multiplier lies in

$[1-\lambda\alpha, 1+\lambda\alpha]$. Compared with using the unclipped factor $1/m_{i,k}$ alone, the numerator $\bar{m}_i$ restores the reweighted component's overall coefficient scale.

If a length factor is clipped, Eqs. (14) and (15) need not hold exactly. Under the parameter ranges in Eqs. (6)–(8), $\rho_{i,k}$ and $w_{i,k}$ remain positive and bounded. Hence $g_{i,k}$ is positive and bounded, and $\widetilde{A}_{i,k}$ preserves the sign of A_i, including $\widetilde{A}_{i,k} = 0$ when $A_i = 0$.