ANCHORCREDIT: OUTPUT-ANCHORED CREDIT ASSIGNMENT FOR STRUCTURED GENERATION WITH MULTIPLE VERIFIABLE REWARDS

Anonymous authors
Paper under double-blind review

ABSTRACT

In Reinforcement Learning with Verifiable Rewards (RLVR), Group Relative Policy Optimization (GRPO) and related methods typically compute group-relative advantages from response rewards and broadcast the same advantage to every generated token in a response to guide policy updates. However, we identify a systematic problem of cross-component credit misassignment in tasks with multiple verifiable results, such as joint multi-attribute prediction: low rewards for some components can penalize a well-performing component and its associated reasoning. To address this problem, we propose **AnchorCredit**. The method uses the policy model's attention from output anchors to preceding reasoning sentences to estimate sentence–component associations and assign component-specific advantages to the relevant reasoning, without additional relevance annotations or a separate relevance discriminator. To support the study of credit assignment with multiple verifiable rewards, we construct ProductAttr, a benchmark of 7,200 real product image–text samples. Experiments show that AnchorCredit outperforms existing baselines on this benchmark. Experiments on the public Nutrition5k dataset further support the method's effectiveness in this setting. The complete code and dataset will be released upon acceptance.

1 INTRODUCTION

Reinforcement Learning with Verifiable Rewards (RLVR) has substantially improved the reasoning capabilities of large language models on automatically verifiable tasks such as mathematics and coding (Shao et al., 2024; DeepSeek-AI, 2025). Existing methods typically evaluate an entire response with a scalar reward. However, many tasks require a model to generate reasoning followed by a structured output containing multiple verifiable output components in a single response. These components may represent product attributes or dialogue slots (Yang et al., 2021; Rastogi et al., 2020). For example, a product understanding task may require jointly predicting several attributes, such as weight, packaged dimensions, and foldability, from product images and text. We study this setting, in which each output component can be verified separately.

When Group Relative Policy Optimization (GRPO) is applied to these tasks, the rewards for the components of each response are weighted and summed to obtain a single response-level reward. These total rewards are standardized across responses sampled for the same prompt to obtain response-level advantages (Shao et al., 2024). The same response-level advantage is then used to weight the policy-loss contribution of every generated token in that response. However, a shared response-level advantage can penalize a component whose own advantage is positive. In Figure 1(a), errors in weight and dimensions result in a negative response-level advantage, causing GRPO to penalize the correct foldability prediction and its associated reasoning despite a positive foldability-specific advantage. We refer to this mismatch as **cross-component credit misassignment**.

Although each output component can be scored separately, these scores do not specify which preceding reasoning sentences should receive its learning signal. Recent work has explored alternatives to GRPO for multi-reward optimization. GDPO (Liu et al., 2026b) normalizes rewards separately before aggregation, while GD^{2}PO (Liu et al., 2026a) filters conflicting rollouts. However, both retain response-level advantages and leave this attribution problem unresolved. RTT (Xu et al., 2026)

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

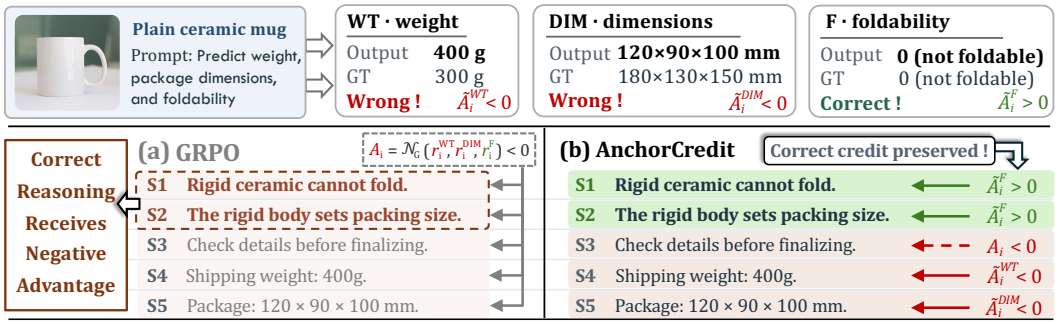


Figure 1: Cross-component credit misassignment and AnchorCredit. (a) A shared negative advantage penalizes correct foldability reasoning. (b) AnchorCredit assigns component-specific credit to associated sentences. Signs are shown before final whitening.

maps rubric feedback to individual tokens through a relevance discriminator, but training this discriminator requires additional token-level relevance annotations and a separate training stage.

To address these limitations, we propose **AnchorCredit**, a plug-in module for component-specific credit assignment. Because each generated output value corresponds to a known component, we use its tokens as anchors. The policy model's attention from these anchors to preceding reasoning sentences provides an estimate of how strongly each sentence is associated with the corresponding component. These associations guide the assignment of component-specific advantages to the relevant sentences (Figure 1(b)), without additional relevance annotations or a separate relevance discriminator. The resulting token-level credit can be incorporated into the policy update of an existing training algorithm.

We introduce ProductAttr, a benchmark of 7,200 product image–text samples for joint prediction of continuous and categorical attributes. Adding AnchorCredit to GRPO improves mean performance on all three target attributes in both ProductAttr and Nutrition5k (Thames et al., 2021). On ProductAttr, the improvements in weight RMAE, dimension RMAE, and foldability F1 are 5.74, 2.80, and 1.76 percentage points, respectively. These gains extend to dialogue state tracking on Schema-Guided Dialogue (SGD; Rastogi et al., 2020), where AnchorCredit improves Joint Goal Accuracy over GRPO by 6.00 percentage points. Gains with DAPO and GDPO show that the benefits extend beyond GRPO, while improvements on larger dense and MoE models further support applicability across model configurations.

In summary, we identify cross-component credit misassignment, where a shared response-level advantage can oppose a component-specific learning signal. To address this mismatch, we propose AnchorCredit, a plug-in module that uses attention from output anchors to assign component-specific credit to reasoning sentences. We also construct ProductAttr with fixed splits and component-wise verifiers, and evaluate the module on multi-attribute prediction and dialogue state tracking.

2 Related Work

Reinforcement learning with verifiable rewards. RLVR uses automatically verifiable rewards for tasks such as mathematics and coding (Shao et al., 2024; DeepSeek-AI, 2025). GRPO computes group-relative advantages without a separately trained value model (Shao et al., 2024); DAPO improves training through asymmetric clipping, dynamic sampling, and a token-level loss (Yu et al., 2025). Both construct advantages from response-level rewards. AnchorCredit instead assigns component-specific credit to relevant positions while retaining the underlying algorithm's sampling and policy-update mechanisms.

Fine-grained credit assignment. Process reward models learn from human or automatically constructed step labels (Lightman et al., 2024; Wang et al., 2024). Other approaches estimate local credit through progress toward success, prefix rollouts, or segment-level advantages (Setlur et al., 2025; Kazemnejad et al., 2025; Guo et al., 2025). Attention-based approaches redistribute response

rewards through reward-model attention (Chan et al., 2024), amplify advantages at selected positions (Li et al., 2025), or weight token losses by attention flow toward the answer, as in FlowTracer (Dong et al., 2026). For multiple feedback signals, RTT maps rubric feedback to token rewards using a separately trained relevance discriminator (Xu et al., 2026); SRaR uses an LLM judge to associate rubric items with reasoning steps (Xie et al., 2026); BAE assigns objective-specific advantages to predefined text blocks (Pavlenko et al., 2026). AnchorCredit instead uses generated output values and the policy model's attention to estimate sentence–component associations and assign component-specific credit, without predefined reasoning blocks, relevance annotations, or a separate relevance model.

Multidimensional rewards and structured prediction. In multi-reward language-model training, GDPO standardizes each reward within the group before aggregating the advantages (Liu et al., 2026b). GD^{2}PO additionally filters and reweights rollouts and queries based on conflicting reward-advantage signs (Liu et al., 2026a). Both retain response-level signals; AnchorCredit complements reward aggregation by assigning component-specific feedback to relevant content positions. MAVE, visual physical-attribute datasets, and Nutrition5k support product attribute extraction, physical-attribute prediction, and nutrition estimation (Yang et al., 2021; Zhai et al., 2024; Thames et al., 2021). Their original settings emphasize supervised prediction; we study how reinforcement learning assigns multiple attribute feedback signals within a generated response.

3 Problem Formulation

We consider tasks in which a model generates reasoning followed by a structured output with K separately verifiable components. For an input x, the policy π_θ generates a group of responses. Let r_i^k denote the reward assigned by the verifier to component k of response i.

GRPO-style methods aggregate component rewards into a scalar reward and standardize it within the group to obtain a response-level advantage. This advantage measures overall performance but does not distinguish the relative performance of each component. Following GDPO (Liu et al., 2026b), we standardize each component's rewards separately within the group. We use the resulting component-specific advantage to measure how a response performs on that component relative to other responses in the group:

$$A_i = \mathcal{N}_G\big(\textstyle\sum_{k=1}^{K} \omega_k r_i^k\big), \qquad A_i^k = \mathcal{N}_G\big(r_i^k\big), \quad k = 1, \ldots, K. \tag{1}$$

Here, ω_k denotes the predefined task weight for component k, and $\mathcal{N}_G$ denotes standardization across responses sampled for the same input. A_i measures relative overall performance.

GRPO applies the same response-level advantage A_i to every generated token (Shao et al., 2024). When $A_i A_i^k < 0$, this shared signal opposes the component-specific advantage, formalizing the cross-component credit misassignment illustrated in Figure 1(a).

To assign component-specific credit to reasoning sentences, we need to identify which sentences are relevant to each component. For component k, we use the tokens of its generated value as **output anchors** and denote their positions in response i by S_i^k. However, which preceding reasoning sentences are relevant to each component is not specified in advance.

We therefore study how to use output anchors to associate reasoning sentences with verifiable output components and assign credit to those sentences based on the corresponding component-specific advantages. For sentences associated with multiple components, we also need to determine how to combine their advantages.

4 AnchorCredit: Output-Anchored Credit Assignment

AnchorCredit constructs token-level credit in three steps: computing component-specific advantages, estimating sentence–component associations using output anchors, and applying association gating and sign-aware combination to obtain sentence credit (Figure 2). Anchor tokens receive the advantage of their corresponding component directly, while the tokens in each reasoning sentence receive that sentence's credit.

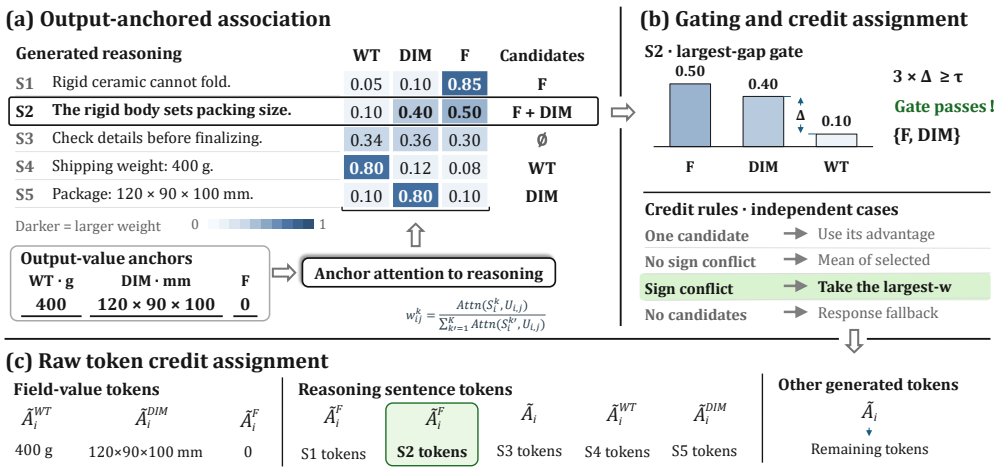


Figure 2: Output-anchored credit assignment. (a) Attention from output-value anchors estimates sentence–component associations. (b) Largest-gap gating selects candidate components; the highlighted example uses the conflicting-sign rule. (c) Component-specific and fallback advantages are assembled into raw token credit before global whitening. Association weights are illustrative.

4.1 From Component-Specific Rewards to Component-Specific Advantages

AnchorCredit starts from the component-specific group-relative advantages A_i^k defined in Equation (1), computed by standardizing the rewards for each component separately across responses to the same input.

Before routing, we center and scale the advantages. We denote this operation by Whiten and apply it separately to the advantages for each component and to the response-level advantages:

$$\widetilde{A}_i^k = \mathrm{Whiten}(A^k)_i, \qquad \widetilde{A}_i = \mathrm{Whiten}(A)_i. \tag{2}$$

Centering may change the signs of the original advantages. All subsequent decisions about matching or conflicting signs use the calibrated advantages $\widetilde{A}_i^k$.

The calibrated advantage $\widetilde{A}_i^k$ specifies the sign and magnitude of credit for component k. We next estimate which reasoning sentences should receive credit from each component. Positions that are not assigned component-specific credit retain $\widetilde{A}_i$ before final whitening.

4.2 Attention-Based Sentence–Component Association Estimation

We use the anchor-token positions S_i^k defined in Section 3 to estimate which reasoning sentences are associated with component k.

Because output anchors follow the reasoning text, we use their attention to preceding tokens to estimate sentence–component associations. We divide the reasoning text of response i into nonempty sentences using boundaries such as sentence-final punctuation and blank lines. We denote sentence j by $U_{i,j}$ and estimate associations at the sentence level. Table 15 in Appendix D.2 compares the end-to-end results of sentence-level and token-level routing; the appendix gives the sentence-splitting rules.

Let $\alpha_{st}^{\ell h}$ denote the attention from anchor token s to preceding reasoning token t in head h of layer ℓ. Let $\mathcal{L}$ be the full-attention layers from which explicit attention weights are extracted, and let H be the number of heads in each such layer. Here, $U_{i,j}$ also denotes the token positions of sentence j. We average equally over these layers, heads, anchor tokens, and sentence tokens:

$$\mathrm{Attn}(S_i^k, U_{i,j}) = \frac{1}{|\mathcal{L}|H|S_i^k||U_{i,j}|} \sum_{\ell\in\mathcal{L}} \sum_{h=1}^{H} \sum_{s\in S_i^k} \sum_{t\in U_{i,j}} \alpha_{st}^{\ell h}. \tag{3}$$

For each reasoning sentence, we normalize these attention scores across components to obtain the association weights:

$$w_{ij}^{k} = \frac{\mathrm{Attn}(S_i^k, U_{i,j})}{\sum_{k'=1}^{K} \mathrm{Attn}(S_i^{k'}, U_{i,j})}, \qquad k = 1, \ldots, K. \tag{4}$$

The weight w_{ij}^{k} estimates the relative association between sentence j and component k. The schema determines which component an anchor belongs to. We use attention as an association estimate for credit routing (Jain & Wallace, 2019; Wiegreffe & Pinter, 2019). Nearly uniform weights do not distinguish a preferred component, while multiple selected components may have advantages with opposing signs. We therefore use a gate to select candidate components before combining their advantages.

The separation between anchor-token credit and reasoning routing is specified in the credit-assembly step below.

4.3 Candidate-Component Gating and Signed Credit Assignment

The preceding steps provide an advantage $\widetilde{A}_i^k$ for each component and association weights w_{ij}^{k} for each reasoning sentence. AnchorCredit uses these weights to select candidate components and combines their advantages according to their signs. The resulting sentence credit is assigned to every token in that sentence, while the remaining tokens are handled in the credit-assembly step.

Candidate selection using the normalized largest gap. A top-1 rule selects one component even when the association weights are nearly uniform. Instead, we select candidates using the largest gap between adjacent sorted weights. Let $w_{ij}^{(1)} \geq \cdots \geq w_{ij}^{(K)}$ denote the weights in descending order, and define

$$\begin{aligned} g_m &= w_{ij}^{(m)} - w_{ij}^{(m+1)}, && m = 1, \ldots, K-1, \\ m^* &= \min\{m : g_m = \max_{1 \leq q < K} g_q\}, && \widetilde{g}_{ij} = K g_{m^*}. \end{aligned} \tag{5}$$

Only when $\widetilde{g}_{ij} \geq \tau$ does AnchorCredit retain components before the largest gap whose association weights strictly exceed the uniform baseline $1/K$. Otherwise, the candidate set is empty:

$$\mathcal{E}_{ij} = \begin{cases} \{k : \mathrm{rank}(w_{ij}^{k}) \leq m^*,\ w_{ij}^{k} > 1/K\}, & \widetilde{g}_{ij} \geq \tau, \\ \varnothing, & \widetilde{g}_{ij} < \tau. \end{cases} \tag{6}$$

Multiplying the gap by K expresses it relative to the uniform weight $1/K$. Thus, a fixed threshold τ corresponds to a raw-gap threshold of τ/K; we use $\tau = 0.5$ in the main configuration. The equations assume that all components are available. When some are unavailable, both attention normalization and gating use only the available components, whose count replaces K. When fewer than two components are available, reasoning tokens receive the calibrated response-level advantage. The gate selects a proper subset of the available components; with two available components, it therefore selects one component or falls back.

Same-sign aggregation and conflicting-sign handling. The candidate set determines which components contribute to the sentence credit. We use the candidate advantage directly when only one component is selected. For multiple candidates, we average their advantages if the set does not contain both a strictly positive and a strictly negative advantage. If both signs are present, averaging may cancel the advantages; we instead use the advantage of the candidate with the largest estimated association weight. The resulting credit for sentence $U_{i,j}$ is

$$c_{ij} = \begin{cases} \widetilde{A}_i^k, & \mathcal{E}_{ij} = \{k\}, \\ |\mathcal{E}_{ij}|^{-1} \displaystyle\sum_{k \in \mathcal{E}_{ij}} \widetilde{A}_i^k, & |\mathcal{E}_{ij}| > 1 \text{ and no conflicting-sign pair exists}, \\ \widetilde{A}_i^{k^*}, \quad k^* = \arg\max_{k \in \mathcal{E}_{ij}} w_{ij}^{k}, & |\mathcal{E}_{ij}| > 1 \text{ and a conflicting-sign pair exists}, \\ \widetilde{A}_i, & \mathcal{E}_{ij} = \varnothing. \end{cases} \tag{7}$$

Estimated association weights determine candidate selection, whereas the selected component-specific advantages determine the sentence credit. If the candidate set is empty, the sentence receives the calibrated response-level advantage $\widetilde{A}_i$.

Credit for three token types and the policy objective. After sentence credit has been computed, raw credit is assembled for three types of generated tokens: (1) field-value tokens, whose component identity is specified by the schema, receive $\widetilde{A}_i^k$; (2) reasoning tokens receive the credit c_{ij} of their sentence; and (3) all remaining tokens retain the response-level advantage $\widetilde{A}_i$:

$$C_{it}^{\mathrm{raw}} = \begin{cases} \widetilde{A}_i^k, & t \in S_i^k, \\ c_{ij}, & t \in U_{i,j}, \\ \widetilde{A}_i, & \text{otherwise.} \end{cases} \tag{8}$$

Here, S_i^k contains only successfully located field-value positions for which a component-specific advantage is available. Reasoning positions that cannot be routed use $\widetilde{A}_i$. Failure to route a reasoning sentence does not undo field-value credit that has already been assigned. If neither field-value assignment nor reasoning routing is performed, all valid positions initially receive $\widetilde{A}_i$, before final whitening and clipping.

Routing changes how many tokens receive each advantage. To standardize the assembled token credit, we jointly whiten it, then clip it to $[-3, 3]$. Let $m_{it} \in \{0, 1\}$ denote the valid-response-token mask:

$$\widehat{C}_{it} = m_{it}\,\mathrm{clip}(\mathrm{Whiten}(C^{\mathrm{raw}})_{it}, -3, 3)\,. \tag{9}$$

Sign-aware advantage combination precedes global token-credit whitening.

We use the final credit in a clipped policy objective with token-level importance ratios. For token t in response i, $\rho_{it}(\theta)$ is the ratio of its probabilities under the current and old policies, conditioned on the same input and preceding tokens. We replace the response-level advantage at each position with $\widehat{C}_{it}$:

$$\mathcal{L}_{\mathrm{AC}}(\theta) = -\mathbb{E}_{i,t}\Big[\min\Big(\rho_{it}(\theta)\widehat{C}_{it}, \mathrm{clip}(\rho_{it}(\theta), 1-\epsilon, 1+\epsilon)\widehat{C}_{it}\Big)\Big]\,. \tag{10}$$

Equation (10) shows the standard symmetric clipping form. Each implementation retains the underlying algorithm's sampling procedure, token-level importance ratios, clipping settings, and KL regularization when used. Replacing the response-level advantage with position-specific credit changes the surrogate objective.

Implementation. Our FlashAttention-based training stack does not expose the attention weights needed for routing. We therefore use an additional forward pass without gradients to extract anchor attention and aggregate it by reasoning sentence; no additional backward pass is required. In the profiled training interval, the attribution prepass takes 19.85 seconds per step, accounting for 13.45% of AnchorCredit's total step time. This separate pass is not intrinsic to AnchorCredit: a modified attention kernel could accumulate the required anchor–sentence statistics during an existing policy forward pass, avoiding both a separate pass and materialization of the full attention matrix. We plan to implement this kernel-level optimization and evaluate its runtime and memory overhead in future work.

5 Experiments

We evaluate whether AnchorCredit improves multi-component prediction, whether its benefits extend across training frameworks, and how its credit-assignment choices affect performance. We then examine sentence–component associations independently of downstream prediction metrics.

5.1 Experimental Setup

Tasks and rewards. ProductAttr contains 7,200 product image–text samples, split into 6,400 training and 800 evaluation examples. It requires predicting packaged weight, dimensions, and foldability, with separate rewards for the three attribute groups. On Nutrition5k (Thames et al., 2021), we predict fat, carbohydrates, and protein from one overhead image per dish, using 2,755 training and 507 evaluation dishes from the official RGB splits. We also evaluate dialogue state tracking on Schema-Guided Dialogue (SGD; Rastogi et al., 2020). Data preparation, output fields, and evaluation conventions are detailed in Appendix A.

Models and comparisons. Our main experiments use Qwen3.5-4B and compare the initial model without task-specific RL (No-RL), GRPO (Shao et al., 2024), GDPO (Liu et al., 2026b), an adapted GD^{2}PO (Liu et al., 2026a), and RTT (Xu et al., 2026). Unless specified otherwise, AnchorCredit is added to GRPO. In the tables, Ours denotes the AnchorCredit module. Within each task, comparisons use the same data and evaluation protocol; host-framework integrations use matched configurations within each pair. ProductAttr training uses 600 updates with eight responses per prompt. Full training settings and the integration interface appear in Appendix B.

Metrics and statistics. We report ratio-of-sums RMAE for continuous attributes, F1 for foldability, and Joint/Average Goal Accuracy (JGA/AGA) for SGD. Unless stated otherwise, means and sample standard deviations use three decoding seeds for a fixed training checkpoint. Independent training-seed replication is reported separately. Paired confidence intervals and their resampling units are specified in Appendix C.3.

5.2 Main Results

Table 1: Main results on ProductAttr and Nutrition5k (%). Each method uses one fixed training checkpoint. Avg. is the arithmetic mean of fat, carbohydrate, and protein RMAE, following the macronutrient summary in Thames et al. (2021). The statistical protocol is specified in Section 5.1.

	ProductAttr-Eval			Nutrition5k (RMAE ↓)			
Method	Weight RMAE ↓	Dimensions RMAE ↓	Fold F1 ↑	Avg. ↓	Fat	Carbs	Protein
No-RL	112.80±16.01	81.57±19.34	65.49±3.77	56.33±2.20	60.14±3.49	55.70±1.19	53.15±2.04
GRPO	30.86±0.30	22.48±0.76	89.55±0.89	35.02±0.39	40.10±0.77	32.37±0.41	32.59±0.11
GDPO	26.23±0.31	20.54±0.07	88.29±0.53	38.72±2.09	44.27±2.18	35.12±2.22	36.79±2.05
GD^{2}PO	27.38±0.60	21.67±0.52	78.24±0.26	39.68±0.48	44.91±0.34	35.74±0.58	38.40±0.76
RTT	29.87±0.14	23.52±0.85	90.91±0.34	37.44±0.53	42.27±1.11	32.64±0.59	37.41±0.50
GRPO + Ours	**25.12**±0.34	**19.68**±0.06	**91.31**±0.42	**33.05**±0.36	**37.10**±0.57	**31.47**±0.93	**30.59**±0.63

Multi-attribute prediction. Adding AnchorCredit to GRPO improves mean performance on all three target attributes in both benchmarks (Table 1). On ProductAttr, weight and dimension RMAE decrease from 30.86 to 25.12 and from 22.48 to 19.68, respectively, while Fold F1 increases from 89.55 to 91.31. On Nutrition5k, the improvements extend to all three nutrients, reducing average RMAE from 35.02 to 33.05. GRPO + AnchorCredit also achieves better mean performance than RTT on all six target attributes, without training a separate relevance discriminator. These results show that assigning component-specific credit can improve joint prediction beyond broadcasting a shared response-level advantage. Paired bootstrap intervals support the continuous-attribute gains for the fixed ProductAttr checkpoints (Appendix C.3).

Dialogue state tracking. The gains also extend to structured outputs in a different domain. On SGD, adding AnchorCredit increases JGA from 75.31 to 81.31 and AGA from 94.48 to 96.48, gains of 6.00 and 2.00 percentage points (Table 2). Here, credit is assigned to dialogue-state components rather than physical attributes. This task-specific training evaluation broadens the evidence beyond image-based attribute prediction.

Table 2: Dialogue state tracking on SGD (%; higher is better). Mean and sample standard deviation use three decoding seeds per fixed training checkpoint. The full comparison, including No-RL, is in Table 11 in Appendix C.1.

Method	JGA ↑	AGA ↑
GRPO	75.31±0.06	94.48±0.02
GRPO + Ours	**81.31**±0.11	**96.48**±0.01

Replication across training seeds. The ProductAttr comparison remains favorable when summarized over three independently trained checkpoints per method. Mean weight/dimension RMAE is 26.10/20.16 with AnchorCredit, compared with 28.35/21.21 for GRPO, and mean Fold F1 rises from 90.00 to 91.72 (Table 10 in Appendix C.1). This replication complements the decoding-seed variation reported in the main table.

5.3 Compatibility Across Training Frameworks and Models

Table 3: Compatibility across training frameworks and models on ProductAttr (%). Panel (a) integrates AnchorCredit with DAPO and GDPO on Qwen3.5-4B; panel (b) integrates it with GRPO on larger dense and MoE models. "+ Ours" adds AnchorCredit to the baseline; each pair uses matched configurations within its setting. Results follow Section 5.1.

Setting	Method	Weight RMAE ↓	Dimensions RMAE ↓	Fold F1 ↑
(a) Training frameworks: Qwen3.5-4B				
DAPO	Original	27.96±1.07	21.48±0.03	90.82±0.13
	+ Ours	**26.13**±0.42	**20.07**±0.03	**91.00**±0.21
GDPO	Original	26.23±0.31	20.54±0.07	88.29±0.53
	+ Ours	**25.02**±0.41	**19.72**±0.18	**91.88**±0.43
(b) Model scale and architecture: GRPO				
Qwen3.5-9B	GRPO	28.17±0.65	20.93±0.24	91.48±0.66
(dense)	+ Ours	**23.49**±0.53	**19.22**±0.07	**92.79**±0.47
Qwen3.5-35B-A3B	GRPO	31.99±1.27	21.35±0.12	90.44±0.31
(MoE)	+ Ours	**25.06**±0.98	**18.94**±0.17	**92.25**±0.57

Integration with DAPO and GDPO. To test whether the gains depend on GRPO, we integrate AnchorCredit with DAPO (Yu et al., 2025) and GDPO, retaining each framework's sampling and policy-update rules. Both pairs improve all three mean ProductAttr metrics (Table 3, panel a). With DAPO, weight and dimension RMAE decrease by 1.83 and 1.41 percentage points. With GDPO, they decrease by 1.21 and 0.82 points, while Fold F1 increases by 3.59 points. These paired comparisons support AnchorCredit as a reusable credit-assignment module within the evaluated frameworks. In particular, the GDPO integration shows that assigning credit to relevant positions can complement component-wise reward normalization.

Model and output extensions. The gains also extend to larger Qwen dense and MoE models: adding AnchorCredit improves all three mean metrics for Qwen3.5-9B and Qwen3.5-35B-A3B (Table 3, panel b). The five-attribute ProductAttr extension improves the mean scores of four attributes (Table 13 in Appendix C.2). The cross-family comparison appears in Table 14 in Appendix D.1.

5.4 Credit-Assignment Analysis

Where should component-specific credit be assigned? We compare assigning component-specific credit only to output values, only to reasoning sentences, or to both (Table 4). Reasoning-only yields lower weight and dimension errors than Value-only, indicating that the allocation of credit within reasoning affects predictive performance. Assigning credit to both regions gives the lowest continuous errors in this comparison, while Reasoning-only gives the highest Fold F1. The lower continuous errors relative to Value-only support assigning component-specific feedback to reasoning sentences.

Table 4: Credit-assignment regions on ProductAttr-Eval (%). Results use three evaluation seeds and the statistical protocol in Section 5.1.

Variant	Weight RMAE ↓	Dimensions RMAE ↓	Fold F1 ↑
GRPO	30.86±0.30	22.48±0.76	89.55±0.89
Value-only	30.80±1.12	21.88±0.09	89.39±0.57
Reasoning-only	25.27±0.72	19.90±0.29	**92.54**±0.53
AnchorCredit	**25.12**±0.34	**19.68**±0.06	91.31±0.42

Routing and credit construction. Sentence-level routing reduces weight and dimension RMAE relative to token-level routing, from 27.01/20.40 to 25.12/19.68 (Table 15 in Appendix D.2). Additional gating and sign-combination comparisons appear in Table 16 in Appendix D.2; the variants exhibit different trade-offs across attributes.

5.5 Offline Association Analysis

Evaluation protocol. We evaluate associations on responses generated by the Qwen3.5-4B base model and GPT-5, independently of our GRPO/AnchorCredit training runs. Each source contains eight responses for each of 128 training products. Both sources are replayed through Qwen3.5-4B to extract attention and gradient signals; GPT-generated responses do not provide GPT's internal attention. We evaluate agreement with sentence–component reference labels generated by GPT-5.6 Terra.

Table 5: Association diagnostics on ProductAttr against Terra reference labels (%). Multi-label association micro-F1 on Base and GPT responses. Both response sources are replayed through Qwen3.5-4B for attention and gradient extraction. Random controls average ten seeds.

Multi-label association *Micro-F1* ↑		
Method	Base	GPT
Matched random	34.14	33.44
Keyword heuristic	61.09	64.43
Gradient gate	38.05	48.74
AnchorCredit	**76.50**	**73.62**

Association quality. AnchorCredit achieves higher micro-F1 than keyword, gradient, and matched-random controls on both response sources (Table 5). Relative to keyword matching, F1 improves by 15.41 points on Base responses and 9.19 points on GPT responses. On Base responses, this reflects higher recall at lower precision; on GPT responses, both increase. Thus, output-anchor attention captures associations beyond explicit keyword overlap on these two sources. Appendix E.1 reports precision, recall, coverage, and an ungated Top-1 variant, which routes every sentence. The ungated variant has higher F1, highlighting the coverage trade-off introduced by candidate gating.

Primary-component identification. We additionally evaluate Top-1 primary-component prediction on GPT-generated responses (Table 6). The gated variant achieves 92.29% precision, compared with 90.21% for keyword matching, 41.92% for gradient attribution, and 32.63% for random assignment. Without gating, AnchorCredit achieves 84.28% precision. Gating retains the same Top-1 choice on accepted sentences, selecting more reliable assignments through abstention. Reference-label construction and the scoring protocol are detailed in Appendix E.1.

Table 6: Top-1 primary-component precision on GPT responses (%). Precision measures the fraction of predicted assignments matching the reference; predictions on empty-reference sentences count as errors.

Primary-component prediction *GPT responses*	
Method	Precision ↑
Random	32.63
Matched random	32.93
Keyword	90.21
Gradient	41.92
AnchorCredit (gated)	**92.29**
AnchorCredit (ungated)	84.28

6 Conclusion and Limitations

Conclusion. We identified cross-component credit misassignment in structured generation, where a shared response-level advantage can oppose a component-specific learning signal. To address this mismatch, we introduced AnchorCredit, a plug-in module that uses the policy model's attention from output-value anchors to assign component-specific credit to reasoning sentences, without additional relevance annotations or a separate relevance discriminator. To evaluate this approach, we constructed ProductAttr, a benchmark of 7,200 product image–text samples with component-wise verifiers. Experiments on this benchmark, Nutrition5k, and dialogue state tracking show improvements over GRPO. Further gains with DAPO and GDPO and on larger dense and MoE models support the module's applicability across training algorithms and model configurations.

Limitations. AnchorCredit requires separately verifiable components and locatable output values. Our current FlashAttention integration uses an additional forward pass to obtain anchor attention; kernel-level extraction could eliminate this separate pass (Section 4.3). Future work will implement this optimization and evaluate its runtime and memory overhead.

Statement on the Use of AI

We used generative AI tools for literature retrieval, experimental-method review, code implementation and debugging, statistical-analysis scripts, interpretation of results, and drafting and editing manuscript text and structure. We used GPT-5 to generate responses and GPT-5.6 Terra to annotate sentence–component associations for the offline evaluation. Qwen and GPT models assisted with

ProductAttr screening and semantic-label checks, with human review of disputed cases. Product images and text come from real product records. Reported numerical results are computed from saved model outputs using fixed evaluation scripts, and important code changes are checked through tests or artifact-level verification. The authors are responsible for the final text, code, results, citations, and conclusions.

Reproducibility Statement

The problem formulation and core algorithm are described in Section 3 and the method section. Data splits, output schemas, training configurations, statistical tests, and completed ablations are documented in the corresponding appendix sections.

Ethics Statement

Nutrition5k uses public data. Permission to release ProductAttr images, text, labels, fixed splits, and accompanying code has been confirmed. ProductAttr has not yet been publicly released; we plan to release these materials upon acceptance, with the applicable licenses and usage terms. The release will exclude credentials, internal infrastructure details, and sensitive fields.

References

Alex James Chan, Hao Sun, Samuel Holt, and Mihaela van der Schaar. Dense reward for free in reinforcement learning from human feedback. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 6136–6154, 2024. URL https://proceedings.mlr.press/v235/chan24a.html.

DeepSeek-AI. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Zhichen Dong, Yang Li, Yuhan Sun, Weixun Wang, Yijia Luo, Zinian Peng, Taiheng Ye, Chao Yang, Wenbo Su, Yu Cheng, Bo Zheng, and Junchi Yan. How does reasoning flow? tracing attention-induced information flow for targeted RL in LLMs. *arXiv preprint arXiv:2606.10646*, 2026.

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*, 2025.

Sarthak Jain and Byron C. Wallace. Attention is not explanation. In *Proceedings of NAACL-HLT*, pp. 3543–3556, 2019. doi: 10.18653/v1/N19-1357.

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 *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 29557–29590, 2025.

Yang Li, Zhichen Dong, Yuhan Sun, Weixun Wang, Shaopan Xiong, Yijia Luo, Jiashun Liu, Han Lu, Jiamang Wang, Wenbo Su, Bo Zheng, and Junchi Yan. Attention illuminates LLM reasoning: The preplan-and-anchor rhythm enables fine-grained policy optimization. *arXiv preprint arXiv:2510.13554*, 2025.

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. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/aca97732e30bcf1303bc22ac3924fd16-Paper-Conference.pdf.

Haotian Liu, Yihao Liu, Jingwei Ni, Siyuan Huang, Xinpeng Liu, Pengyu Cheng, Jiajun Song, Ruijin Ding, Junfeng Li, Zhechao Yu, Mengyu Zhou, Hongteng Xu, Xiaoxi Jiang, and Guanjun Jiang. GD^{2}PO: Mitigating multi-reward conflicts via group-dynamic reward-decoupled policy optimization. *arXiv preprint arXiv:2606.16771*, 2026a.

Shih-Yang Liu, Xin Dong, Ximing Lu, Shizhe Diao, Peter Belcak, Mingjie Liu, Min-Hung Chen, Hongxu Yin, Yu-Chiang Frank Wang, Kwang-Ting Cheng, Yejin Choi, Jan Kautz, and Pavlo Molchanov. GDPO: Group reward–decoupled normalization policy optimization for multi-reward RL optimization. *arXiv preprint arXiv:2601.05242*, 2026b.

Kirill Pavlenko, Alexander Golubev, Simon Karasik, and Boris Yangel. Blockwise advantage estimation for multi-objective RL with verifiable rewards. *arXiv preprint arXiv:2602.10231*, 2026. URL https://arxiv.org/abs/2602.10231.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 8689–8696, 2020. doi: 10.1609/aaai.v34i05.6394. URL https://ojs.aaai.org/index.php/AAAI/article/view/6394.

Amrith Setlur, Chirag Nagpal, Adam Fisch, Xinyang Geng, Jacob Eisenstein, Rishabh Agarwal, Alekh Agarwal, Jonathan Berant, and Aviral Kumar. Rewarding progress: Scaling automated process verifiers for LLM reasoning. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=A6Y7AqlzLW.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, 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.

Quin Thames, Arjun Karpur, Wade Norris, Fangting Xia, Liviu Panait, Tobias Weyand, and Jack Sim. Nutrition5k: Towards automatic nutritional understanding of generic food. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8903–8911, 2021.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu 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*, pp. 9426–9439, 2024. doi: 10.18653/v1/2024.acl-long.510.

Sarah Wiegreffe and Yuval Pinter. Attention is not not explanation. In *Proceedings of EMNLP-IJCNLP*, pp. 11–20, 2019. doi: 10.18653/v1/D19-1002.

Weichu Xie, Haozhe Zhao, Wenpu Liu, Yongfu Zhu, Liang Chen, Minghao Ye, Zirong Chen, Yuqi Xu, Shuai Dong, Ziyue Wang, Xinbo Xu, Kean Shi, Ruoyu Wu, Xiaoying Zhang, Wenqi Shao, Baobao Chang, Nan Duan, and Jiaqi Wang. Step-wise rubric rewards for LLM reasoning. *arXiv preprint arXiv:2605.17291*, 2026. URL https://arxiv.org/abs/2605.17291.

Tianze Xu, Yanzhao Zheng, Pengrui Lu, Lyumanshan Ye, Yong Wu, Zhentao Zhang, Yuanqiang Yu, Chao Ma, Jihuai Zhu, Pengfei Liu, Baohua Dong, Hangcheng Zhu, Ruohui Huang, and Gang Yu. Rubrics to tokens: Bridging response-level rubrics and token-level rewards in instruction following tasks. In *Conference on Language Modeling*, 2026. URL https://arxiv.org/abs/2604.02795.

Li Yang, Qifan Wang, Zac Yu, Anand Kulkarni, Sumit Sanghai, Bin Shu, Jon Elsas, and Bhargav Kanagal. MAVE: A product dataset for multi-source attribute value extraction. *arXiv preprint arXiv:2112.08663*, 2021.

Qiying Yu, Zheng Zhang, Ruofei Zhu, et al. DAPO: An open-source LLM reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Albert J. Zhai, Yuan Shen, Emily Y. Chen, Gloria X. Wang, Xinlei Wang, Sheng Wang, Kaiyu Guan, and Shenlong Wang. Physical property understanding from language-embedded feature fields. *arXiv preprint arXiv:2404.04242*, 2024.

A Data, Prompts, and Verifiers

A.1 Data Splits and Deduplication

Product sources and labels. ProductAttr contains 7,200 distinct AliExpress products, each represented by one stock-keeping unit (SKU), spanning 26 top-level categories, 100 second-level categories, and 1,075 leaf categories. Product images and text are matched to warehouse records by product and SKU identifiers. Weight and dimensions refer to the packaged product, measured in grams and millimeters, rather than net weight or unpackaged dimensions. For repeated measurements, the maximum recorded values are retained. All physical labels are complete, parseable, and positive.

Foldability indicates whether the SKU can be safely folded or rolled under fulfillment rules that account for packaging and damage risk. The dataset contains 5,040 nonfoldable and 2,160 foldable products. The supplementary seasonality labels indicate pronounced seasonal relevance in temperate Northern Hemisphere markets and may include multiple seasons. Festival labels identify one of seven specific holidays from the product's design or intended use; products without a clear holiday association receive the nonfestive label. Semantic labels are checked against product images, text, and business rules using Qwen and GPT models, with human review of disputed cases.

Construction starts from 100 high-sales categories. Model-assisted screening removes inconsistent product information, image–text mismatches, category errors, and implausible warehouse measurements. Additional candidates are retrieved to improve coverage of rare festival labels. After label review and product-level deduplication, deterministic sampling maintains category coverage and increases representation of rare labels. The resulting label proportions reflect dataset construction rather than the product population.

Input attributes and quality checks. Standard product attributes are obtained from the platform's product-information system. Only item-level standard English attribute names and values are retained, separately from SKU-selection attributes. When constructing the input attribute list, we remove weight, dimensions, foldability, seasonality, and festival fields, including aliases and direct proxies, as well as identifiers and backend metadata. This filtering applies only to the attribute list; natural evidence in titles, descriptions, and images is retained. After cleaning, 6,714 samples have nonempty standard attributes and 486 have empty attribute lists. Automated checks cover field structure, valid enumerations, physical-value validity, and explicit label fields in the attribute list; they do not replace manual quality checks for each product.

The dataset retains the existing seed-42 split of 6,400 training samples and 800 ProductAttr-Eval samples. Before splitting, image URLs, titles, and descriptions undergo Unicode NFKC, case, and whitespace normalization. Samples sharing any nonempty key are connected into the same group. There are 6,310 groups, with at most 70 samples in a group, and none of these exact-duplicate keys overlap across splits. After grouping, distributions are balanced over category, foldability, currency, text-missingness status, and weight, dimension, and volume bins. Deduplication uses normalized exact-match keys.

Images are snapshots of the original product images. All methods share the same preprocessing and data splits. Table 7 summarizes the data splits and release scope.

Table 7: Data splits, deduplication, and release scope.

Dataset	Train	Eval	Total examples	Grouping and release scope
ProductAttr	6,400	800	7,200	Exact-duplicate groups; planned release of images, text, and labels
Nutrition5k	2,755	507	3,262	Official dish IDs; public data

Product benchmark release scope and status. Release of the complete product images, product text, attribute labels, fixed splits, and accompanying code has been confirmed as permissible. These materials will be released upon acceptance, together with documentation, applicable licenses, and download links.

The Nutrition5k-Overhead-507 adaptation uses 2,755/507 dishes from the official RGB train/test dish-ID lists. The two dish-ID sets are disjoint and contain 3,262 dishes in total. Training constructs six deterministic image variants for each training dish, yielding 16,530 training examples. These variants are not counted as independent dishes and never cross the train/test boundary. The adaptation does not introduce a separate validation split. All methods use a fixed 517-step budget and evaluate their final checkpoints on the fixed evaluation set of 507 dishes. At evaluation, each dish uses one overhead RGB image aligned with official metadata by dish ID. This setting is not a direct reproduction of the official multiview 2D main protocol. Final materials will release the complete dish-ID lists, their correspondence to the official splits, the single-image selection rule, and fixed resizing and normalization procedures. The formal task predicts fat, carbohydrates, and protein, reporting RMAE separately for the three attributes. Paired methods use the same dish lists and preprocessing. Dishes with zero-valued targets remain in the fixed evaluation set; their absolute errors contribute to the numerator of the dataset-level ratio-of-sums RMAE defined in Appendix A.3, which does not divide by an individual dish's target.

A.2 Output Fields and Verifiers

The product task is trained with three reward/credit groups: WT (weight), DIM (joint dimensions), and F (foldability). Five output fields do not constitute five independent tasks. Length, width, and height share one attribute reward and one routing group (Table 8).

Table 8: Output fields and the three attribute groups in the main product setting.

Attribute group	JSON field	Unit/type	Verifier
Product WT	`item_weight_cn`	g	Weight scoring
Product DIM	`item_length_cn`, `item_width_cn`, `item_height_cn`	mm	Joint dimension scoring
Product F	`final_fold_flag`	0/1	Label matching

Nutrition5k predicts fat, carbohydrates, and protein, with three separately reported RMAE values.

ProductAttr training rewards. Let p and g denote a prediction and its target. Weight is measured in grams and dimensions in millimeters. For $0 < g < 500$, the weight reward combines a 50-gram bucket match, distance to the target bucket, and a continuous log-ratio score. Define $b(x) = \lceil x/50 \rceil$, the target interval $I_g = [\max\{0, 50(b(g) - 1)\}, 50b(g)]$, and $d(p, I_g)$ as the distance from p to this interval. With $H = \mathbf{1}[p > 0 \text{ and } b(p) = b(g)]$ and $C = \max\{0, 1 - |\ln(p/g)|/\ln 4\}$ for $p > 0$ ($C = 0$ otherwise), the weight reward is

$$r_{\mathrm{WT}} = \begin{cases} 0.60H + 0.25 \exp\left(-\dfrac{d(p, I_g)}{50 \times 0.5}\right) + 0.15C, & 0 < g < 500, \\ \max\left\{0, 1 - \dfrac{|p - g|}{|g| + 10^{-6}}\right\}, & g \geq 500. \end{cases}$$

The three dimension fields contribute to one reward:

$$E_{\mathrm{DIM}} = \frac{\sum_{a \in \{L,W,H\}} |p_a - g_a|}{\sum_{a \in \{L,W,H\}} |g_a| + 10^{-6}}, \qquad r_{\mathrm{DIM}} = \max\{0, 1 - E_{\mathrm{DIM}}\}.$$

This training reward compares the original named axes. The implementation separately constructs the dimension signal used as input to component-specific advantage estimation: its bucket correction sorts the three axes and uses 50-mm buckets, a correction ratio of 0.2, and a correction cap of 0.05. These settings affect `adv_signal_dimensions`; the scalar reward uses the WMAPE-based r_{DIM} above.

Foldability receives reward 1 when prediction and target match after conversion to strings, whitespace stripping, and lowercasing, and 0 otherwise. A missing foldability prediction receives 0. With all labels present, the response reward is $R = 0.4r_{\mathrm{WT}} + 0.4r_{\mathrm{DIM}} + 0.2r_{\mathrm{F}}$. A JSON parse failure gives reward 0 for each labeled component. Missing numerical predictions enter the corresponding reward formula as zero. Components with missing labels are excluded, and the aggregation weights are renormalized over the remaining components. The main configuration disables the additional AnchorCredit premise penalty.

Nutrition5k training rewards. The three active fields are `fat_g`, `carb_g`, and `protein_g`. For positive predictions and targets, the component reward and response reward are

$$r_k = \max\{0, 1 - |\log_2(p_k/g_k)|\}, \qquad R = \frac{r_{\text{fat}} + r_{\text{carb}} + r_{\text{protein}}}{3}.$$

Thus, an exact prediction receives 1, whereas a prediction at half or twice the target receives 0. Missing, nonnumerical, nonfinite, or negative predictions receive component reward 0 and remain in the three-component denominator. A zero prediction receives 0 for a positive target; when the target is zero, only a zero prediction receives 1. JSON parse failures or a key set differing from the three active fields receive format reward 0. Severe repetition sets the response reward and all three component rewards to -0.4. It is triggered by a substantive sentence occurring at least five times, or by 20-character segments that each occur at least three times and together cover at least 25% of the reasoning text. The configuration applies no additional short-reasoning, thinking-tag, or soft length penalty.

A.3 Evaluation Metrics

Ratio-of-sums RMAE for each reported numerical output in either task is

$$R_k = \frac{\sum_{i=1}^{N} |\hat{y}_{ik} - y_{ik}|}{\sum_{i=1}^{N} |y_{ik}|}.$$

For Nutrition5k, k indexes fat, carbohydrates, or protein. A dish with $y_{ik} = 0$ remains in the evaluation set: its absolute error contributes to the numerator, while the dataset-level denominator avoids per-dish division by zero. The numerator and denominator of joint-dimension error R_{DIM} each sum over both samples and the three axes. Fold F1 is reported separately.

A.4 Prompts and Structured Outputs

ProductAttr uses a frozen English task prompt, while product text retains its original language. Each message supplies one product image together with the title, description, top-level/second-level/leaf categories, standard attributes, SKU attributes, and SKU price with currency. Supervision labels are not provided to the model. The complete template will be released with the dataset version. Its output constraint reads:

> First provide your analysis as plain text. Then finish the response with exactly one JSON object containing exactly these five keys:
>
> ```
> {"item_weight_cn":  0.0,
>  "item_length_cn":  0.0,
>  "item_width_cn":  0.0,
>  "item_height_cn":  0.0,
>  "final_fold_flag":  0}
> ```

The example zero values illustrate field types and are not target labels. The template requires positive finite continuous values, Fold in 0/1, no units in numerical values, no Markdown code fence around the JSON, and no content after the final JSON. It also requires the model to distinguish whether numerical values in the description refer to the current SKU, a component, another specification, or shipping packaging, and to use the image to judge applicability. The prompt does not require brief reasoning or enforce `<think>` or `<answer>` tags.

B Training Configurations and Reproducibility Details

Table 9 lists the main ProductAttr training settings. Each method changes only its prescribed advantage or credit construction; host-framework pairs retain their native rules.

Table 9: Training and generation settings. The main comparison uses training seed 42. Core product methods are also trained with seeds 43 and 44; see the multiple-training-seed results table.

Setting	Main ProductAttr setting
Model/initialization	Qwen3.5-4B / Base
Learning rate and KL	10^{-6}; KL loss 2×10^{-3}
clip / loss aggregation	0.20 / seq-mean-token-mean
rollout / effective batch	32 prompts $\times$ 8 = 256
Input/output limits	5,120 / 8,192 tokens
Training T/top-p/top-k	$0.9/0.99/100$
Evaluation sampling	$0.9/0.99/100$
evaluation seeds	42, 43, 44
Update budget/endpoint	600 / checkpoint-599
Checkpoint interval	100 updates and final checkpoint
Hardware	8 A100 GPUs per run

ProductAttr uses AdamW with weight decay 0.01, a 25-step warmup, and a cosine learning-rate schedule. Training uses one PPO epoch, a frozen visual module, and bfloat16 (bf16). The main configuration uses gap threshold $\tau = 0.50$; thresholds of 0.75 and 1.00 are evaluated in Table 12.

AnchorCredit pseudocode. Token credit is constructed in the following order:

1. Standardize rewards separately for each output within each prompt's rollout group to obtain A_i^k, and obtain A_i from the total reward. Whiten these quantities to obtain $\widetilde{A}_i^k$ and $\widetilde{A}_i$.
2. Locate the output anchors S_i^k and assign $\widetilde{A}_i^k$ to their tokens.
3. Split reasoning into sentences $U_{i,j}$, aggregate attention from anchors to each sentence, and normalize over available outputs to obtain w_{ij}^k.
4. Apply the normalized largest-gap gate to the sorted w_{ij}^k to obtain candidate set $\mathcal{E}_{ij}$. Reasoning sentences that fail the gate use $\widetilde{A}_i$.
5. For a single candidate, use its $\widetilde{A}_i^k$. For multiple candidates without conflicting signs, average their advantages. For candidates with conflicting signs, select the output-specific advantage with the largest association weight.
6. Use $\widetilde{A}_i^k$ for the corresponding field-value tokens, c_{ij} for reasoning-sentence tokens, and $\widetilde{A}_i$ for other tokens. Whiten and clip valid tokens again to obtain $\widehat{C}_{it}$, and insert this credit into the underlying clipped policy objective.

Two-stage whitening. The policy surrogate uses group-relative advantage standardization, pre-routing minibatch whitening, fallback, policy-ratio clipping, and post-assembly whole-batch whitening. The first whitening operation separately calibrates the batch scales of each output-advantage column and the response-level advantage. The second jointly centers and scales all valid tokens after field, sentence, and fallback credit have been assembled. Consequently, the second operation also changes final coefficients at unrouted positions. Section 5.4 describes the Value-only versus GRPO field-credit comparison.

Integration across algorithms. The common AnchorCredit interface receives output-specific reward vectors from the same rollout group and the policy statistics required by the underlying policy-optimization algorithm. It returns assembled raw credit C_{it}^{raw} and final credit $\widehat{C}_{it}$. In the main GRPO experiment, $\widehat{C}_{it}$ replaces the response-level advantage while sampling, clipping, and regularization remain unchanged. DAPO can use the same token-credit interface, but its results are interpreted only within separately matched runs. The GDPO integration first standardizes individual rewards according to GDPO's rules and computes the whole-batch mean and standard deviation of the aggregated advantage. Contributions from each output are centered and scaled using that same mean and standard deviation so that their average agrees with GDPO's response-level advantage. The contributions are then assigned to relevant positions and final token whitening is applied. This integration does not recompute a separate batch mean and standard deviation for each output advantage.

Cross-optimizer results are interpreted only within their matched controls; absolute scores do not replace the GRPO-based method comparison in the main text.

Reasoning-sentence segmentation. Sentence boundaries follow sentence-final punctuation and blank lines, excluding decimal points.

Attention aggregation. The implementation equally averages all full-attention layers that can return explicit attention matrices. It further averages over all attention heads, value tokens for the same output, and tokens within the same reasoning sentence. Linear-attention layers in hybrid architectures that do not produce such matrices are excluded. Averaging all available layers introduces no additional layer-selection hyperparameter.

B.1 RTT Adaptation

We adapt RTT (Xu et al., 2026) using a separately trained Qwen3.5-4B discriminator with a scalar token-classification head. It takes an attribute criterion and a generated response as input and outputs a relevance probability for each response token. For ProductAttr, supervision covers weight, joint dimensions, and foldability. The labeling pipeline combines `qwen3-max` relevance spans with deterministic JSON-field spans and negative examples in which the criterion is omitted. It uses 608 responses from 76 policy-training entities, split into 60 training and 16 validation entities, yielding 2,880 training and 768 validation criterion–response pairs. For Nutrition5k, LLM-labeled spans from training-set responses are converted to character-level labels and split by dish. Its discriminator covers five nutrient attributes, while the Macro3 policy queries only fat, carbohydrates, and protein. Discriminator training and validation use neither policy-validation nor test examples.

The discriminator uses binary cross-entropy on labeled response tokens, masking prompts, padding, and invalid labels. The training entry point specifies Qwen3.5-4B initialization, two epochs, learning rate 9×10^{-6}, maximum length 2,048, global batch size 64, and microbatch size 1 per GPU on four GPUs, with seed 42, bf16, ZeRO-2, SDPA, and gradient checkpointing. It is pinned to official RTT revision `b1ab2fba9bec`. During policy training, the discriminator supplies continuous relevance probabilities without binary thresholding. The policy and discriminator check vocabulary token-ID compatibility and reuse the generated token IDs for alignment.

For response i, component k, and token t, let z_{ikt} be the discriminator logit. We compute $q_{ikt} = \mathrm{sigmoid}(z_{ikt})$ and $u_{ikt} = q_{ikt} r_i^k$. The auxiliary signal standardizes u_{ikt} over valid tokens of the same response, separately for each component, and averages over the three components:

$$S_{it} = \frac{1}{3} \sum_{k=1}^{3} \frac{u_{ikt} - \mathrm{mean}_{t'}\, u_{ikt'}}{\sqrt{\mathrm{var}_{t'}\, u_{ikt'} + 10^{-6}}}, \qquad A_{it}^{\mathrm{RTT}} = \mathrm{clip}\big(A_{it}^{\mathrm{GRPO}} + 0.5 S_{it}, -3, 3\big)\,.$$

Here, t' ranges over valid response tokens. Normalization is performed within each response rather than across responses; the resulting signal is added to the GRPO advantage.

B.2 Adapted GD^{2}PO

We extend the response-level conflict filtering in GD^{2}PO (Liu et al., 2026a) to the three task rewards. Within each prompt's group $\mathcal{G}$ of eight responses, we compute

$$a_i^k = \frac{r_i^k - \mathrm{mean}_{j \in \mathcal{G}}\, r_j^k}{\mathrm{std}_{j \in \mathcal{G}}\, r_j^k + 10^{-4}}.$$

ProductAttr weights these values by $(0.4, 0.4, 0.2)$ for weight, dimensions, and foldability; Nutrition5k uses their unweighted sum. The hard-conflict rule masks a response if its valid component advantages include both a value above 10^{-8} and a value below -10^{-8}, ignoring missing components. Let m_i be 1 for a retained response and 0 otherwise. The response-level policy advantage before token normalization is

$$\rho_{\mathcal{G}} = \frac{1}{8} \sum_{j \in \mathcal{G}} m_j, \qquad A_i^{\mathrm{GD^2PO}} = m_i \rho_{\mathcal{G}} \sum_{k=1}^{3} v_k a_i^k,$$

where $v = (0.4, 0.4, 0.2)$ for ProductAttr and $v = (1, 1, 1)$ for Nutrition5k. We broadcast this value to response tokens, whiten only valid tokens of retained responses, and clip to $[-3, 3]$. Masked responses retain zero policy-gradient advantage. A fully filtered group contributes zero policy-gradient advantage; if the entire batch is filtered, whitening is skipped. We do not resample after filtering, so each rollout batch remains 32 prompts with eight responses each. Masking affects the policy-gradient term; the separate KL loss remains active.

C Complete Results and Statistical Tests

C.1 Training-Seed Replication and Dialogue State Tracking

Table 10: ProductAttr summary across three training seeds (%): means are reported.

Method	Weight RMAE ↓	Dimensions RMAE ↓	Fold F1 ↑
GRPO	28.35	21.21	90.00
GRPO + Ours	**26.10**	**20.16**	**91.72**

Table 11: SGD test results (%; higher is better). All methods report the mean and sample standard deviation over decoding seeds 42, 43, and 44 for one fixed model checkpoint per method, not variation across training seeds. Approximately 90% of Base outputs fail the frozen output-conformance checks and remain in the reported denominators; the Base-to-trained gap therefore includes output-format compliance.

Method	Joint Goal Accuracy ↑	Average Goal Accuracy ↑
Base (No-RL)	5.77±0.63	7.54±0.40
GRPO	75.31±0.06	94.48±0.02
+ Ours	81.31±0.11	96.48±0.01

C.2 Supplementary ProductAttr Results

Table 12: Gap comparison on ProductAttr-Eval (%), using fixed endpoints and three evaluation seeds.

Gap	Weight RMAE ↓	Dimensions RMAE ↓	Fold F1 ↑	Valid-output rate ↑
0.50	**25.12**±0.34	**19.68**±0.06	91.31±0.42	99.67±0.14
0.75	27.92±0.14	20.82±0.06	**93.00**±0.48	**100.00**±0.00
1.00	28.77±0.75	21.62±0.18	92.14±0.37	**100.00**±0.00

Larger gap thresholds increase weight and dimension errors but improve Fold F1 relative to the default threshold of 0.50, which is used in the main experiments and component ablations.

FiveTask extension evaluation. To examine credit assignment after adding seasonality and festival attributes, we compare matched GRPO and AnchorCredit on the same ProductAttr-Eval split. Table 13 shows that AnchorCredit reduces weight and dimension RMAE by 3.18 and 2.28 percentage points, respectively, and increases sample-level seasonality Dice by 1.80 percentage points. Fold F1 is approximately unchanged, and festival macro-F1 increases by 0.32 percentage points. Seasonality Dice matches the task's training reward and measures per-product label-set agreement, including the nonseasonal label.

Table 13: ProductAttr FiveTask extension (%), using a fixed single training checkpoint and three evaluation seeds. S-Dice is sample-level seasonal Dice; F-Macro is Macro-F1 over eight holiday classes.

Method	Weight RMAE ↓	Dimensions RMAE ↓	Fold F1 ↑	S-Dice ↑	F-Macro ↑
GRPO	29.36 ± 0.57	21.98 ± 0.38	$\mathbf{91.60} \pm 0.06$	89.43 ± 0.47	96.41 ± 0.30
AnchorCredit	$\mathbf{26.18} \pm 0.43$	$\mathbf{19.70} \pm 0.04$	91.51 ± 0.32	$\mathbf{91.23} \pm 0.23$	$\mathbf{96.73} \pm 0.80$

C.3 Statistical Tests

Table 10 summarizes replication across three ProductAttr training seeds. The intervals below instead quantify evaluation variability for the fixed models trained with seed 42.

For ProductAttr, we use a hierarchical paired bootstrap over evaluation seeds and products, with `item_id+sku_id` as the pairing unit. Each replicate draws three evaluation seeds with replacement and resamples the 800 products, sharing the sampled product indices across methods, metrics, and selected seeds. We recompute the metrics for each sampled seed and average across seeds. We use $B = 20{,}000$ replicates and percentile 95% confidence intervals. Reported standard deviations across evaluation seeds use ddof=1.

For AnchorCredit minus GRPO, the differences in weight RMAE, dimension RMAE, and Fold F1 are -5.74, -2.80, and $+1.76$ percentage points, respectively. Their 95% intervals are $[-8.48, -3.09]$, $[-4.99, -1.69]$, and $[-1.27, 4.85]$. The continuous-attribute improvement intervals exclude zero, whereas the Fold interval includes zero. These intervals do not cover training randomness; other between-method differences are descriptive unless an interval is explicitly reported.

D Additional Baselines, Ablations, and Sensitivity Analysis

D.1 Cross-Family Evaluation

Table 14 reports the Gemma3-4B-IT comparison. GRPO and GRPO + AnchorCredit use the same initialization, data, training budget, and decoding configuration. Larger Qwen dense and MoE results are reported in Table 3, panel b; the MoE model has approximately 35B total parameters and activates approximately 3B per token.

Table 14: Cross-family evaluation on Gemma3-4B-IT using ProductAttr-Eval (%). Means and sample standard deviations use three evaluation seeds for fixed training checkpoints, following Section 5.1.

Method	Weight RMAE ↓	Dimensions RMAE ↓	Fold F1 ↑
GRPO	$\mathbf{51.53} \pm 0.74$	133.86 ± 37.96	55.91 ± 1.63
GRPO + Ours	52.25 ± 2.27	$\mathbf{50.88} \pm 24.86$	$\mathbf{59.15} \pm 1.14$

For Gemma3-4B-IT, AnchorCredit improves dimension RMAE and Fold F1, while weight RMAE increases from 51.53 to 52.25. Dimension errors have large standard deviations across decoding seeds, indicating substantial variability in this configuration.

D.2 Online Routing and Credit-Combination Controls

Routing granularity. Token-route selects candidates and assigns credit separately at each reasoning token, rather than aggregating attention within a sentence. Both variants use gap threshold 0.50 and the same field-value credit, conflict handling, whitening, training budget, and evaluation protocol. Sentence-level routing has lower weight and dimension errors, while Token-route has slightly higher Fold F1.

Table 15: Online routing-granularity comparison on ProductAttr-Eval (%). Both variants assign component-specific credit to final values and reasoning; results use three evaluation seeds.

Variant	Weight RMAE ↓	Dimensions RMAE ↓	Fold F1 ↑
Token-route	27.01±2.01	20.40±0.63	**91.46**±0.76
AnchorCredit (sentence route)	**25.12**±0.34	**19.68**±0.06	91.31±0.42

Gating and conflicting signs. Mean and Conflict fallback retain the default candidate sets and gate, changing only the rule for candidates with opposite-sign advantages. Mean averages their advantages; Conflict fallback restores response-level credit. Default SignGuard instead selects the most strongly associated candidate. Same-sign combinations, field-value credit, and whitening remain unchanged. Hard Top-1 always selects the most strongly associated output and removes the gate, so it does not isolate the effect of SignGuard.

Table 16: Gating and conflict handling on products (%). Results use three evaluation seeds and the statistical protocol in Section 5.1.

Variant	Weight RMAE ↓	Dimensions RMAE ↓	Fold F1 ↑
Mean	25.67±0.16	19.95±0.11	92.07±0.60
Conflict fallback	26.94±1.94	**19.59**±0.13	92.10±1.07
Hard Top-1	25.43±0.08	20.02±0.04	**92.99**±0.22
AnchorCredit	**25.12**±0.34	19.68±0.06	91.31±0.42

The default method has the lowest weight error, Conflict fallback the lowest dimension error, and Hard Top-1 the highest Fold F1 (Table 16).

D.3 Pre-Routing Credit-Normalization Ablation

Table 17 compares the method with and without pre-routing per-output whitening. Results use the same three-evaluation-seed protocol as the main ProductAttr comparison.

Table 17: Pre-routing credit-normalization ablation on ProductAttr-Eval (%).

Variant	Weight RMAE ↓	Dimensions RMAE ↓	Fold F1 ↑
NoPerAttrWhiten	25.66±0.13	**19.45**±0.01	**92.32**±0.46
AnchorCredit	**25.12**±0.34	19.68±0.06	91.31±0.42

Removing pre-routing per-output whitening slightly worsens weight error but improves dimensions and Fold.

E Supplementary Association Analysis

E.1 Model-Reference Association Evaluation

Table 18 reports the full association metrics. The Base and GPT response sets each contain 1,024 responses to 128 ProductAttr training inputs. Base responses yield 43,183 scored sentences from 961 responses; 63 responses have no locatable reasoning sentences and remain in response-level accounting but do not contribute sentence-level labels. All 1,024 GPT responses yield scored sentences, totaling 7,807. Fifty-five Base responses reach the generation limit; this status is separate from the no-sentence category. The reference model is GPT-5.6 Terra (`gpt-5.6-terra-2026-07-09`). It labels complete visible reasoning using the fixed component-labeling protocol. Attention and gradient attribution on both sources are obtained by replay through Qwen3.5-4B.

Table 18: Full model-reference association metrics on Base and GPT responses (%). Precision, recall, and F1 are micro-averaged. Coverage is the percentage of sentences with a nonempty predicted component set. Random matching uses the gated method's per-sentence candidate counts and averages ten random seeds. Top-1 is an ungated attention ablation.

Source	Association method	Precision	Recall	F1	Coverage
Base	Matched random	34.94	33.39	34.14	88.99
	Keyword heuristic	90.78	46.04	61.09	48.66
	Gradient gate	37.29	38.85	38.05	95.91
	Top-1 without gating	79.59	75.41	77.44	100.00
	AnchorCredit	78.28	74.80	76.50	88.99
GPT	Matched random	43.35	27.22	33.44	79.77
	Keyword heuristic	94.88	48.77	64.43	65.19
	Gradient gate	55.01	43.75	48.74	99.83
	Top-1 without gating	94.25	67.31	78.53	100.00
	AnchorCredit	95.41	59.92	73.62	79.77

We use 2,000 paired bootstrap replicates over 128 product clusters per source. The 95% F1 intervals for AnchorCredit are [74.66, 78.27] on Base and [72.28, 74.88] on GPT responses. Differences from keyword matching are +15.41 [14.05, 16.80] and +9.19 [7.82, 10.44]. Differences from ungated Top-1 are -0.94 [-1.46, -0.22] and -4.91 [-5.70, -4.13]. Intervals describe variability over sampled products, not uncertainty in reference labels. Each source is evaluated separately.

Primary-component reference protocol. The GPT response set contains 7,681 relevant sentences and 126 sentences with empty reference labels. Existing single-component labels are retained; for the 2,557 multi-label sentences, Terra selects one primary component from the original reference set. Uncertain primary labels (909 sentences) remain in scoring. Precision is the number of correct primary-component assignments divided by the total number of emitted assignments across all 7,807 sentences. Assignments to empty-reference sentences count as false positives; abstentions do not enter the precision denominator. AnchorCredit (ungated) selects the highest attention score. AnchorCredit (gated) makes the same selection on accepted sentences and otherwise abstains. Their precision values are 84.28% and 92.29%, respectively. Matched random uses the same acceptance mask as the gate and achieves 32.93% precision. Random controls average ten seeds. This single-primary-label diagnostic is distinct from the multi-label evaluation above.

E.2 Nutrition5k Prompt Control

Nutrition5k prompt control. We additionally train GRPO and GRPO + AnchorCredit with an English prompt that removes the original restrictions on the number of checks and revisions, while retaining prediction of total fat, carbohydrates, and protein for the whole dish and the strict three-field JSON output requirement. Both arms use this prompt and evaluate final checkpoint-516 on the same 507 samples with evaluation seeds 42, 43, and 44. Table 19 reports the paired results. This control changes both language and reasoning guidance, so differences cannot be attributed to any one instruction.

Table 19: Paired Nutrition5k results with the new prompt: attribute RMAE (%, lower is better), reported as the mean ± sample standard deviation across three evaluation seeds.

Method	Fat	Carbohydrate	Protein
GRPO	$\mathbf{34.71}\pm0.30$	30.39 ± 0.38	33.01 ± 0.31
GRPO + Ours	36.29 ± 0.75	$\mathbf{29.82}\pm0.68$	$\mathbf{31.96}\pm0.03$