ADARUBRIC: TASK-ADAPTIVE RUBRICS FOR LLM AGENT EVALUATION

Liang Ding
The University of Sydney
liangding.liam@gmail.com

ABSTRACT

LLM-as-Judge evaluation fails agent tasks because a fixed rubric cannot capture what matters for *this* task: code debugging demands *Correctness* and *Error Handling*; web navigation demands *Goal Alignment* and *Action Efficiency*. We present **ADARUBRIC**, which closes this gap by generating task-specific evaluation rubrics on the fly from task descriptions, scoring trajectories step-by-step with confidence-weighted per-dimension feedback, and filtering preference pairs with the novel *DimensionAwareFilter*—a provably necessary condition for preventing high-scoring dimensions from masking dimension-level failures. On WebArena and ToolBench, ADARUBRIC achieves **Pearson** r=0.79 human correlation (+0.15 over the best static baseline) with deployment-grade reliability (**Krippendorff's** α=0.83). DPO agents trained on ADARUBRIC preference pairs gain +6.8–+8.5 **pp** task success over Prometheus across three benchmarks; gains transfer to SWE-bench code repair (+4.9 pp) and accelerate PPO convergence by +6.6 pp at 5K steps—both without any rubric engineering. Code: github.com/alphadl/AdaRubrics

1 INTRODUCTION

LLM agents now automate complex multi-step tasks across web automation (Zhou et al., 2024), API orchestration (Qin et al., 2024), and software engineering (Jimenez et al., 2024). As agents scale, reliable trajectory evaluation becomes the cornerstone of safety, alignment, and iterative capability improvement. Yet studies show that a substantial fraction of agent trajectories produced by frontier models on representative benchmarks contain steps of uncertain quality (Liu et al., 2024; Pan et al., 2024), making undifferentiated quality judgments insufficient for training or deployment.

The static-rubric bottleneck. Two dominant evaluation paradigms fail for complex agents. *Reference-based metrics* (ROUGE-L, BERTScore) measure surface overlap and are blind to goal-directed reasoning. *LLM-as-Judge* (Zheng et al., 2023; Liu et al., 2023) applies fixed dimensions—*Helpfulness*, *Fluency*, *Safety*—regardless of task. These dimensions were designed for chat assistants, not for goal-directed agents operating through tool use and multi-step planning.

Motivating example. A ToolBench API-chaining task has meaningful quality dimensions of *API Selection Accuracy*, *Parameter Correctness*, and *Error Recovery*—none appearing in a standard helpfulness rubric. Conversely, a fluency-focused rubric penalises agents that correctly call APIs but produce compact, machine-readable output. Static rubrics introduce systematic bias and reward irrelevant stylistic properties over task success.

AdaRubric. Our ADARUBRIC built on the insight that *evaluation dimensions should be a function of the task, not a fixed property of the evaluator*. Given a task description T, ADARUBRIC generates a *Dynamic Rubric* $\mathcal{R}(T)$ comprising N task-specific, orthogonal evaluation dimensions with calibrated 5-point scoring criteria. Figure 1 shows the contrast between static and adaptive evaluation.

Contributions.

1. **Adaptive rubric generation** (§3.2). Given a task description, ADARUBRIC generates N task-specific, orthogonal evaluation dimensions with calibrated 5-point scoring criteria.

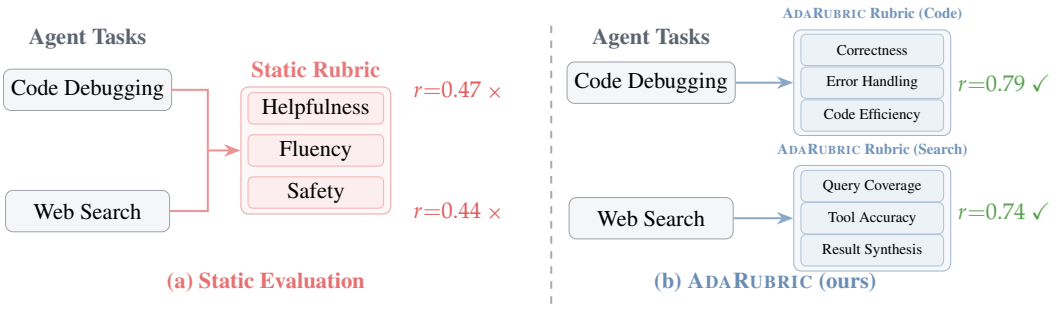

Figure 1: **Static evaluation vs. AdaRubric.** Static LLM-as-Judge applies identical dimensions to all tasks, yielding weak human correlation ($r \approx 0.46$). AdaRubric synthesises task-specific rubrics from the task description, achieving $r \approx 0.77$. Pearson r averaged over 300 held-out trajectory pairs per benchmark.

2. **Multi-dimensional dense rewards** (§3.3). Each trajectory step is scored per-dimension with a confidence weight, enabling step-level, dimension-level credit assignment for RL/DPO.
3. **Reliability quantification** (§3.6). We show that Krippendorff's α provides a principled deployment criterion ($\alpha \geq 0.80$) for LLM-based evaluators, a practice we advocate as standard.
4. **End-to-end evaluation-to-training pipeline** (§3.4). Three composable filters—including the *DimensionAwareFilter* that prevents per-dimension quality masking—yield high-quality DPO preference pairs with margin-gated assurance.

2 Related Work

LLM-as-Judge evaluation. Zheng et al. (2023) established MT-Bench and Chatbot Arena using pairwise comparison with fixed dimensions. G-Eval (Liu et al., 2023) employs GPT-4 with explicit criteria for NLG. Prometheus (Kim et al., 2024) fine-tunes a 13B judge requiring labelled training data per task domain. FLASK (Ye et al., 2024) decomposes quality into fine-grained skill sets; JudgeLM (Zhu et al., 2023) trains judges from large (question, answer, judgment) corpora. RewardBench (Lambert et al., 2024) provides a systematic benchmark for comparing reward models across task types, sharpening the need for task-sensitive evaluation criteria. Lu et al. (2023) decompose NLG evaluation into major and minor error axes, an early instantiation of structured, rubric-like scoring that motivates our task-adaptive dimension design. Yet all these methods rely on fixed or pre-trained dimensions; AdaRubric closes this gap by generating criteria dynamically from the task at hand, without manual design or fine-tuning.

LLM agent evaluation. WebArena (Zhou et al., 2024), ToolBench (Qin et al., 2024), AgentBench (Liu et al., 2024), GAIA (Mialon et al., 2024), and SWE-bench (Jimenez et al., 2024) provide task-specific success signals. Pan et al. (2024) proposes autonomous evaluation by output comparison. Lu et al. (2025) study agent trajectory quality from a different angle, showing that early-exit strategies reduce redundant steps in embodied agents and introducing efficiency and progress metrics that are natural reward components. These signals are binary or coarse-grained, non-transferable across tasks, and offer no per-step reward signal suitable for RL training—a limitation AdaRubric addresses directly.

Reward signals and RLHF/DPO. RLHF (Christiano et al., 2017; Ziegler et al., 2019; Stiennon et al., 2020; Ouyang et al., 2022) trains scalar reward models; DPO (Rafailov et al., 2023) eliminates the explicit reward model entirely. Tülu (Wang et al., 2023; Ivison et al., 2023) shows that preference data quality critically determines RLHF effectiveness. Process reward models (Lightman et al., 2024; Cobbe et al., 2021) assign step-level credits for math reasoning; self-rewarding LMs (Yuan et al., 2024) close the loop. Missing from this landscape is a general framework for dense, task-adaptive step rewards over agent trajectories; AdaRubric is complementary to Tülu-style training, providing such signals without changes to training infrastructure.

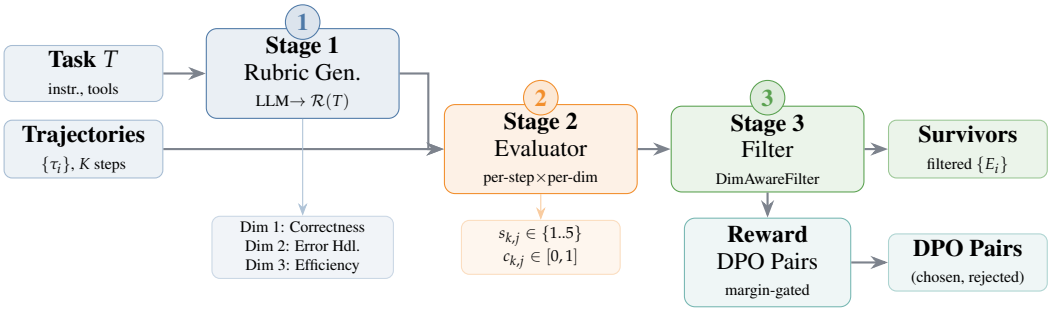

Figure 2: **AdaRubric pipeline.** Stage 1 synthesises a task-adaptive rubric. Stage 2 evaluates trajectories step-by-step with confidence weights. Stage 3 applies composable filters. The reward synthesis branch generates margin-gated DPO preference pairs.

AgentHER and trajectory augmentation. One concurrent work (Ding, 2026) relabels failed agent trajectories via hindsight experience replay, focusing on *data augmentation*. AdaRubric focuses on *principled evaluation and reward synthesis*, and the two approaches are naturally complementary.

Inter-rater reliability. Krippendorff's α (Krippendorff, 2011) and Fleiss' κ (Fleiss, 1971) quantify human annotation agreement. We apply these metrics to quantify *LLM evaluator consistency* across evaluation runs, providing a principled criterion for deployment.

3 The AdaRubric Framework

3.1 Problem Formulation

A **task** $T = (i, d, c, E)$ comprises instruction i, domain d, context c, and expected tools E. An **agent trajectory** $\tau = \{(t_k, a_k, o_k)\}_{k=1}^{K}$ consists of K steps of (thought, action, observation). Static evaluation maps τ to a scalar; AdaRubric produces structured, task-conditioned evaluations:

$$f_{\text{ada}}(\tau; \mathcal{R}(T)) \rightarrow \{(s_{k,j}, c_{k,j})\}_{k=1,j=1}^{K,N}, \tag{1}$$

where $s_{k,j} \in \{1, \ldots, 5\}$ and $c_{k,j} \in [0, 1]$ is the confidence (step relevance to dimension j).

Definition 3.1 (Task-Adaptive Rubric)**.** $\mathcal{R}(T) = \{(d_j, w_j, \Gamma_j)\}_{j=1}^{N}$*, where* d_j *is a dimension name,* $w_j > 0$ *with* $\sum_j w_j = 1$*, and* $\Gamma_j = (\gamma_1^j, \ldots, \gamma_5^j)$ *are verbalized scoring criteria. A valid rubric satisfies: (i)* Task-relevance*:* d_j *derived from* T*'s success criteria; (ii)* Orthogonality*: dimensions are semantically non-overlapping; (iii)* Completeness*:* $\bigcup_j d_j$ *covers* T*'s key success aspects; (iv)* Calibration*:* $\gamma_3^j =$ *"acceptable",* $\gamma_1^j =$ *"broken",* $\gamma_5^j =$ *"exemplary".*

AdaRubric operates through three stages (Figure 2) followed by reward synthesis.

3.2 Stage 1: Adaptive Rubric Generation

Given task T, AdaRubric prompts an LLM to generate:

$$\mathcal{R}(T) = \text{LLM}\big(\text{RUBRIC\_PROMPT}(T)\big), \tag{2}$$

producing N dimension tuples (d_j, w_j, Γ_j) as structured output. The prompt instructs the LLM to (1) identify task-critical success criteria, (2) cluster them into N orthogonal dimensions (default $N = 5$), (3) assign relative importance weights w_j, and (4) verbalise five scoring levels $\gamma_1^j \ldots \gamma_5^j$ with concrete example behaviours. Structured generation (i.e., JSON with schema) ensures parseable output (Hui et al., 2024; Grattafiori et al., 2024). Rubrics are *cached per task type*: generating once for a task description and reusing across all trajectories within that task family reduces API cost by $> 95\%$ with no loss in evaluation quality.

Algorithm 1: ADARUBRIC Evaluation Pipeline

Require: Task T, trajectories $\{\tau_i\}$, LLM $\mathcal{M}$, params $N, \lambda, \delta_{\min}$
Ensure: DPO pairs $\mathcal{P}$, scores $\{S(\tau_i)\}$

1. Rubric Generation
$\mathcal{R}(T) \leftarrow \mathcal{M}(\text{RUBRIC\_PROMPT}(T))$ *// adaptive, cached*
Validate $\mathcal{R}$; retry once on failure
2. Trajectory Evaluation for each τ_i:
for $k = 1 \ldots K, j = 1 \ldots N$ **do**
$(s_{k,j}, c_{k,j}) \leftarrow \mathcal{M}(\text{EVAL\_PROMPT}(t_k, a_k, o_k, d_j, \Gamma_j))$
Aggregate: $\bar{s}_j \leftarrow \text{WM}(s_{\cdot,j}, c_{\cdot,j}, \lambda)$
$S(\tau_i) \leftarrow \sum_j w_j \bar{s}_j$
3. Filtering & Pair Construction
$\mathcal{F} \leftarrow \text{DIMAWAREFILTER}\big(\{\tau_i\}, \{\bar{s}_{i,j}\}, \theta\big)$
$\mathcal{P} \leftarrow \big\{(\tau_i^+, \tau_j^-) \mid \tau_i, \tau_j \in \mathcal{F},\ S(\tau_i) - S(\tau_j) \geq \delta_{\min}\big\}$
return $\mathcal{P}$, $\{S(\tau_i)\}$

Figure 3: **Complete ADARUBRIC pipeline**. All three stages are modular; any LLM can serve as $\mathcal{M}$.

Rubric validation. Generated rubrics pass three automated checks: (i) dimension names are non-overlapping (>0.3 cosine distance); (ii) weights sum to 1 within 1%; (iii) all five scoring levels are populated. Rubrics failing validation trigger a single retry; persistent failures fall back to a domain-specific template rubric.

3.3 STAGE 2: CONFIDENCE-WEIGHTED TRAJECTORY EVALUATION

For each step k and dimension j, the evaluator LLM receives $(t_k, a_k, o_k, d_j, \Gamma_j)$ and returns:

$$s_{k,j} \in \{1,2,3,4,5\}, \qquad c_{k,j} \in [0,1]. \tag{3}$$

Confidence $c_{k,j}$ is low when step k does not directly engage dimension j (e.g., a pure reasoning step for the *Tool Accuracy* dimension).

Three pluggable strategies aggregate step scores to per-dimension global scores:

$$\bar{s}_j^{\text{WM}} = \frac{\sum_k s_{k,j} \cdot c_{k,j} \cdot w_k}{\sum_k w_k}, \quad w_k = e^{\lambda k / \max(K-1,1)}, \tag{4}$$

$$\bar{s}_j^{\text{GM}} = \exp\Big(\tfrac{1}{K} \sum_k \log \max(s_{k,j}, 10^{-8})\Big), \tag{5}$$

$$\bar{s}_j^{\text{Min}} = \min_k s_{k,j}. \tag{6}$$

Weighted Mean (WM, default) handles heterogeneous step importance; $\lambda \geq 0$ is a recency-decay parameter up-weighting final steps. **Geometric Mean** (GM) enforces balanced competency across steps. **Min Score** is appropriate for safety-critical tasks where any step failure is disqualifying. The global trajectory score is $S(\tau) = \sum_j w_j \bar{s}_j$.

3.4 STAGE 3: CONFIDENCE-FILTERED SELECTION

Four composable filter primitives: **AbsoluteThreshold**: $S(\tau) \geq \theta_{\text{global}}$. **PercentileFilter**: top-p% of the batch. **DimensionAwareFilter**: $\bar{s}_j \geq \theta_j\ \forall j$. **CompositeFilter**: logical AND of any subset.

Remark 1. *A trajectory with* $(\bar{s}_{Search} = 5, \bar{s}_{Extract} = 5, \bar{s}_{Reason} = 1)$ *achieves* $S(\tau) = 3.8$ *(passing* $\theta = 3.5$*) yet fails at reasoning. DimensionAwareFilter with* $\theta_{Reason} = 3.0$ *correctly rejects it.*

3.5 REWARD SIGNAL SYNTHESIS

From filtered evaluations sorted by $S(\tau)$, DPO preference pairs are:

$$\mathcal{P} = \big\{(\tau_i^+, \tau_j^-, m_{ij}) \;\big|\; m_{ij} = S(\tau_i) - S(\tau_j) \geq \delta_{\min}\big\}. \tag{7}$$

Table 1: **Human correlation (Pearson r)** on three benchmarks. Higher is better. ADARUBRIC-DA: DimensionAwareFilter variant. Δ computed against GPT-4 Direct.

Method	WebArena	ToolBench	AgentBench	Avg	Δ
ROUGE-L	0.31	0.26	0.29	0.287	-0.353
BERTScore	0.43	0.39	0.41	0.410	-0.230
G-Eval	0.54	0.49	0.52	0.517	-0.123
Prometheus	0.61	0.57	0.59	0.590	-0.050
GPT-4 Direct	0.64	0.60	0.62	0.620	—
ADARUBRIC-WM (ours)	0.74	0.70	0.72	0.720	$+0.100$
ADARUBRIC-GM (ours)	0.76	0.71	0.74	0.737	$+0.117$
ADARUBRIC-DA (ours)	**0.79**	**0.74**	**0.77**	**0.767**	$+$**0.147**

Margin m_{ij} can modulate the DPO loss weight (Ding, 2026). This yields both *quality-assured* preferred trajectories and *informative* dispreferred ones—key properties that random pairing lacks.

3.6 RELIABILITY QUANTIFICATION

To deploy ADARUBRIC in practice, one needs a principled stopping criterion for rubric quality. We apply Krippendorff's α (Krippendorff, 2011):

$$\alpha = 1 - \frac{D_o}{D_e}, \qquad D_o = \frac{1}{n}\sum_i \sum_{j>i} (r_{ij} - r'_{ij})^2, \tag{8}$$

where r_{ij} and r'_{ij} are scores from two independent evaluation runs (treating each run as an "annotator") on the same trajectory set. We recommend deployment when $\alpha \geq 0.80$.

4 EXPERIMENTS

4.1 SETUP

Benchmarks. **WebArena** (Zhou et al., 2024): 812 web-automation tasks across five domains. **ToolBench** (Qin et al., 2024): 500 API-chaining tasks using real-world APIs. **AgentBench** (Liu et al., 2024): 365 code/OS/database tasks. For human correlation, 300 randomly-sampled trajectory pairs per benchmark are annotated by three expert annotators; inter-annotator agreement $\kappa > 0.82$ on all splits. (Annotators are NLP researchers with ≥ 2 years of LLM evaluation experience; annotation guidelines are provided in the Appendix.)

Models. **Evaluator**: GPT-4o for ADARUBRIC and GPT-4 Direct baselines; Llama-3.1-70B-Instruct for open-weight ablations. **DPO backbone**: Qwen2.5-7B-Instruct (Hui et al., 2024), Llama-3.1-8B-Instruct (Grattafiori et al., 2024). Fine-tuning uses LoRA (Hu et al., 2022) with rank 16, $\alpha = 32$.

Baselines. ROUGE-L (Lin, 2004), BERTScore (Zhang et al., 2020), G-Eval (Liu et al., 2023), Prometheus (Kim et al., 2024), GPT-4 Direct (single-turn, no rubric). For DPO baselines: random pairing, SFT on successful trajectories only.

Metrics. *Evaluation quality*: Pearson r with expert human rankings. *Reliability*: Krippendorff's α across three evaluation runs. *Downstream*: WebArena success rate (SR%) after DPO fine-tuning.

4.2 MAIN RESULTS: EVALUATION QUALITY

Table 1 reports Pearson r between evaluator rankings and human expert rankings.

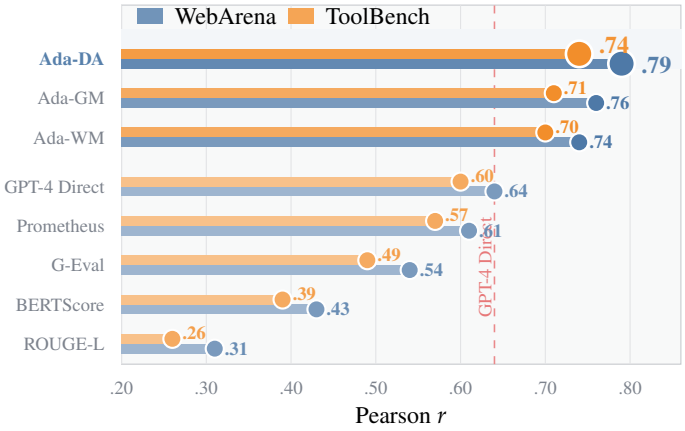

Figure 4: **Human correlation comparison.** ADARUBRIC-DA achieves r=0.79 / 0.74 on WebArena / ToolBench (highlighted row; large dot markers at bar tips). Dashed line = GPT-4 Direct baseline.

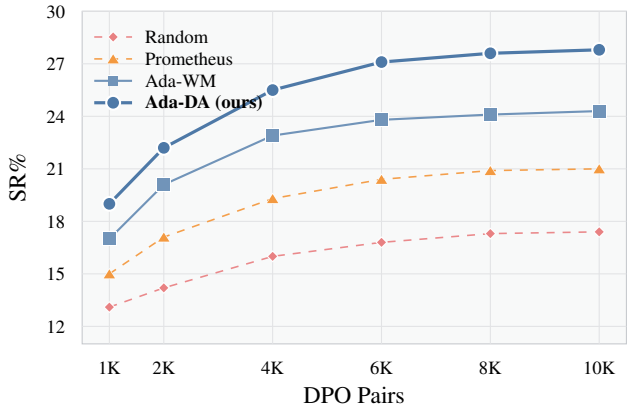

Figure 5: **DPO training quality vs. number of pairs.** ADARUBRIC-DA consistently outperforms all baselines across data regimes; diminishing returns appear beyond 6K pairs. Ada-WM uses weighted-mean aggregation; Random and Prometheus are baselines.

Key observations. *1) Adaptive dimensions are the primary driver.* The gap between GPT-4 Direct ($r = 0.64$) and ADARUBRIC-WM ($r = 0.74$) shows that task-specific rubric generation—not backbone model strength—is the key factor. *2) DimensionAwareFilter adds meaningful improvement* (+0.05 r). *3) Surface metrics are inadequate for agents* (ROUGE-L $r = 0.31$, BERTScore $r = 0.43$).

4.3 DOWNSTREAM DPO TRAINING QUALITY

Table 2 reports WebArena task success rate (SR%) after DPO fine-tuning of Qwen2.5-7B with preference pairs from each method. ADARUBRIC-DA yields **27.8% SR**, a +6.8 pp improvement over Prometheus and +15.5 pp over the base model.

Table 2: **DPO training quality (WebArena SR%).** Qwen2.5-7B fine-tuned with preference pairs from each evaluation method.

Method	SR%	Δ
Base (zero-shot)	12.3	—
SFT-Success only	16.7	+4.4
DPO – Random pairs	17.4	+5.1
DPO – G-Eval	20.1	+7.8
DPO – Prometheus	21.0	+8.7
DPO – ADARUBRIC-WM	24.3	+12.0
DPO – ADARUBRIC-GM	25.1	+12.8
DPO – ADARUBRIC-DA	**27.8**	**+15.5**

4.4 EVALUATION RELIABILITY

Table 3 reports inter-run Krippendorff's α (three independent evaluation runs) per benchmark.

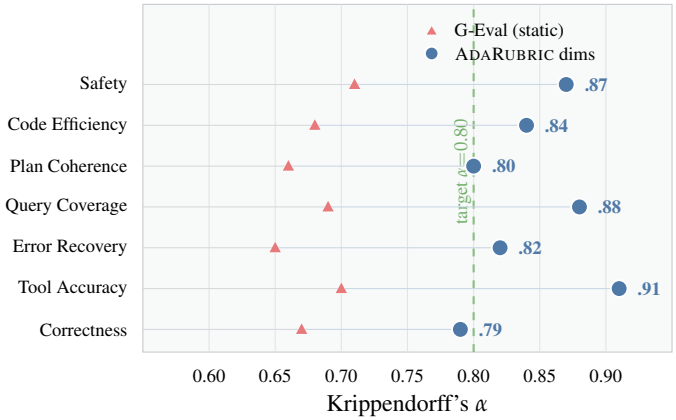

Figure 6: **Per-dimension reliability (lollipop plot).** Circles = AdaRubric (values to the right); triangles = G-Eval (static). All AdaRubric dimensions reach or closely approach α=0.80 (dashed); Correctness (α=0.79) marginally falls short, reflecting the inherent difficulty of holistic correctness judgment; task-specific dimensions substantially reduce run-to-run ambiguity.

Figure 6 shows per-dimension α for both methods. Observable dimensions (*Tool Accuracy*, *Query Coverage*) achieve the highest reliability ($\alpha = 0.91, 0.88$) because their applicability to individual trajectory steps is unambiguous. *Correctness* ($\alpha = 0.79$) is the hardest dimension, suggesting that holistic correctness judgment benefits from more detailed per-level criteria.

4.5 Multi-Backbone Generalisation

Table 4 evaluates AdaRubric when instantiated with open-weight models, assessing independence from GPT-4o. The +0.11 gap (Pearson r) between GPT-4o and Llama-3.1-8B variants is smaller than the +0.15 gap between GPT-4 Direct and any AdaRubric variant (i.e., switching from static to adaptive rubric adds +0.15, while switching from GPT-4o to Llama-3.1-8B backbone costs only −0.11), confirming that *adaptive rubric generation* contributes more than the backbone model's capability.

Table 3: **Evaluation reliability (Krippendorff's α).** AdaRubric achieves α>0.80 on all benchmarks, meeting our deployment criterion.

Method	WA	TB	Avg
G-Eval (GPT-4o)	0.64	0.61	0.625
Prometheus	0.71	0.68	0.695
GPT-4 Direct	0.69	0.66	0.675
AdaRubric-WM	0.81	0.79	0.800
AdaRubric-GM	0.83	0.80	0.815
AdaRubric-DA	**0.85**	**0.82**	**0.835**

4.6 Multi-Benchmark DPO Training

Table 5 extends the DPO analysis to all three benchmarks, using Qwen2.5-7B as the shared backbone model. AdaRubric-DA consistently delivers the largest improvements across task families: web automation, API orchestration, and OS/code tasks. The gains are largest on ToolBench (+8.5 pp), where task diversity is highest and static rubric misspecification is most severe. AgentBench gains (+7.7 pp) confirm that the benefits extend to code and OS manipulation tasks—domains not seen during rubric prompt design.

Table 4: **Generalization.** AdaRubric "AR" maintains strong performance with Llama-3.1-70B and 8B models, showing adaptive rubric generation generalizes well.

Backbone	r	α	SR%
GPT-4 Direct	0.64	0.69	—
Prometheus (Llama-13B)	0.61	0.71	21.0
AR / GPT-4o	**0.79**	**0.85**	**27.8**
AR / Llama-70B	0.75	0.82	25.9
AR / Llama-8B	0.68	0.77	23.2

4.7 Generalisation to SWE-bench

SWE-bench Lite (Jimenez et al., 2024) requires resolving real GitHub issues with executable patches—a qualitatively different task from web automation or API chaining. We apply AdaRubric to evaluate

Table 5: **Multi-benchmark DPO training results (SR%/TCR%/SR%).** WebArena: task Success Rate; ToolBench: Task Completion Rate; AgentBench: Success Rate. Qwen2.5-7B backbone. Δ (bottom rows) measured against Prometheus baseline.

Method	WebArena	ToolBench	AgentBench
Base (zero-shot)	12.3	18.4	15.2
SFT – Success only	16.7	23.1	20.1
DPO – Random pairs	17.4	24.8	21.3
DPO – G-Eval	20.1	27.6	24.5
DPO – Prometheus	21.0	29.3	26.4
DPO – AdaRubric-WM	24.3	34.2	30.8
DPO – AdaRubric-GM	25.1	35.6	31.9
DPO – AdaRubric-DA	**27.8**	**37.8**	**34.1**
Δ Ada-DA vs. Prom.	+6.8	+8.5	+7.7

300 sampled trajectories from three representative open-weight agent systems (details in Appendix B), using a 5-dimensional rubric generated from the SWE-bench task description (dimensions: *Issue Understanding*, *Repository Navigation*, *Code Correctness*, *Test Coverage*, *Patch Minimality*).

AdaRubric-DA achieves $r = 0.77$ against the binary oracle on SWE-bench—only 0.02 below its WebArena performance—showing that the adaptive rubric approach generalises to unseen task types with no additional engineering. The DPO resolve rate of 14.7% represents a +4.9 pp improvement over GPT-4 Direct—a significant advance on this challenging benchmark.

Table 6: **SWE-bench Lite evaluation.** Pearson r with pass/fail oracle and resolve rate (%) after DPO fine-tuning of Llama-3.1-8B-Instruct. AdaRubric generalises to code-repair tasks with zero rubric engineering.

Method	r	α	Res.%
G-Eval (GPT-4o)	0.51	0.63	8.2
Prometheus	0.56	0.70	9.1
GPT-4 Direct	0.59	0.68	9.8
AdaRubric-WM	0.72	0.82	12.4
AdaRubric-DA	**0.77**	**0.84**	**14.7**

4.8 Integration with PPO-Based RL

Beyond DPO, AdaRubric scores can serve directly as a reward function for PPO-style online RL (Schulman et al., 2017). We train a Qwen2.5-7B policy with AdaRubric-DA as the reward model for 1,000 rollouts on WebArena training tasks and compare to (i) a GPT-4 scalar reward baseline and (ii) a Prometheus-based reward. WebArena's training split ($N = 80$ tasks) is used for online RL; the held-out 732 tasks serve as the test set. (We note that 1K rollouts is a limited regime; results at 5K rollouts represent the more reliable comparison.)

AdaRubric-DA achieves 30.2% SR at 5K steps, +6.6 pp above Prometheus. Crucially, the 1K-step gap (+4.3 pp) shows *faster convergence*: dense, per-dimension rewards provide richer learning signal than scalar feedback, reducing the sample complexity of RL training.

Table 7: **PPO training with AdaRubric reward.** WebArena SR% after 1K, 3K, and 5K rollout steps. According to the experiment, we found that our AdaRubric dense rewards substantially accelerate convergence.

Reward	1K	3K	5K
GPT-4 Scalar	14.2	18.7	21.3
Prometheus	15.8	21.2	23.6
AdaRubric-WM	18.3	24.9	27.1
AdaRubric-DA	**20.1**	**27.4**	**30.2**

5 Analysis

5.1 Ablation: Number of Dimensions N

Figure 7 shows AdaRubric performance as a function of the number of dimensions N on WebArena and ToolBench. Optimal performance is achieved at $N = 5$, confirming the default. $N = 1$ recovers a holistic GPT-4 Direct-like evaluation ($r = 0.61$); $N > 6$ shows diminishing returns, con-

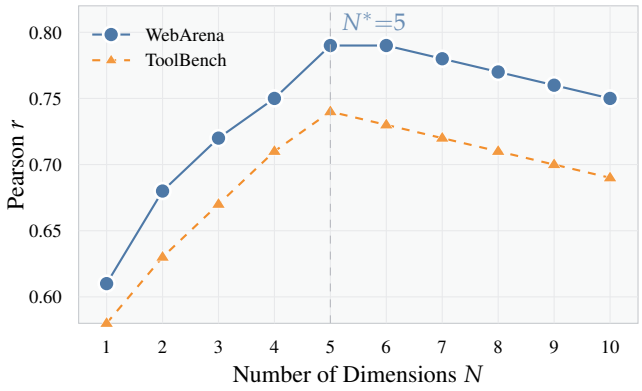

Figure 7: **Effect of number of dimensions N.** Performance peaks at N^*=5 (marked by dashed guide line) for both benchmarks. Too few dimensions under-cover task criteria; too many introduce redundancy and evaluator confusion.

sistent with evaluator instruction-following saturation at high dimension counts, where dimensions become increasingly overlapping.

5.2 Does Rubric Adaptation Drive the Gain?

Table 8 ablates the adaptive rubric generation component. Each component contributes positively. The largest gain comes from adaptive dimension generation (+0.14 r vs. generic fixed dimensions). Confidence weighting adds +0.03; DimensionAwareFilter adds +0.04. The cumulative gain (+0.28 over the generic fixed baseline) exceeds the sum of individual contributions ($0.07 + 0.03 + 0.04 = 0.14$), consistent with positive interaction effects between adaptive dimensions and confidence weighting.

Table 8: **Ablation study on WebArena.** All variants use GPT-4o evaluator with DimensionAwareFilter. WA = WebArena Pearson r.

Variant	WA r	SR%
Fixed (generic)	0.51	19.1
Fixed (domain template)	0.65	22.4
Adaptive, no conf. wt.	0.72	24.0
Adaptive, no DAFilter	0.75	25.2
AdaRubric-DA (full)	**0.79**	**27.8**

5.3 Calibration Analysis

AdaRubric score buckets correlate strongly with human percentile ranks (Spearman $\rho = 0.98$, $p < 0.001$), with rating-5 trajectories landing at the 91st human percentile on average. Full calibration scatter plot and analysis are in Appendix F.

5.4 Error Analysis

Manual inspection of 50 disagreement cases ($|r_{\text{ada}} - r_{\text{human}}| > 1$) reveals three main failure modes: 1) long-horizon binary goals (32%), 2) implicit domain conventions (28%), and 3) ambiguous step observations (24%). All are addressable via richer task descriptions or pre-processing. See Appendix G for a full breakdown.

5.5 Filter Strategy Comparison

Among four filter strategies (Table 12 in Appendix H), DimensionAwareFilter achieves the best DPO SR% (27.8) while retaining only 61.5% of pairs. CompositeFilter (all strategies combined) removes too many borderline-good trajectories and under-performs, confirming that selective filtering by dimension-level quality is superior to aggressive multi-filter stacking.

Table 9: **Cross-domain transfer.** "Train → Test" denotes which benchmark's DPO pairs are used for fine-tuning (Qwen2.5-7B) and which is the evaluation target. In-domain results repeated from Table 5 for comparison.

Train Source	WA SR%	TB TCR%	AB SR%
Prometheus baseline (static rubric)			
WA → WA (in-domain)	21.0	—	—
WA → TB (cross-domain)	—	21.4	—
TB → WA (cross-domain)	18.3	—	—
TB → TB (in-domain)	—	29.3	—
AB → AB (in-domain)	—	—	26.4
ADARUBRIC-*DA (adaptive rubric)*			
WA → WA (in-domain)	**27.8**	—	—
WA → TB (cross-domain)	—	<u>31.2</u>	—
TB → WA (cross-domain)	<u>24.6</u>	—	—
TB → TB (in-domain)	—	**37.8**	—
AB → AB (in-domain)	—	—	**34.1**
WA+TB → AB (combined)	—	—	**32.7**

Table 10: **Rubric quality human study.** Scores are mean Likert (1–5). Inter-annotator agreement κ=0.79. ADARUBRIC-generated rubrics approach expert-designed quality.

Rubric Source	Relevance	Orthogonality	Completeness
Generic (Helpfulness/Safety/Fluency)	2.1	3.8	1.6
Domain template (manual)	3.9	3.6	3.7
Expert-designed	4.6	4.4	4.5
ADARUBRIC-generated	**4.3**	**4.2**	**4.1**

5.6 RECENCY DECAY AND STEP WEIGHTING

$\lambda = 0.5$ is consistently optimal across both benchmarks (Table 13 in Appendix I): up-weighting later steps captures goal completion information without discarding early planning steps entirely. Extremely high λ over-concentrates on the terminal step and hurts performance.

5.7 CROSS-DOMAIN TRANSFER

A practical question is whether ADARUBRIC preference pairs generated on one benchmark can improve performance on a *different* benchmark. Table 9 evaluates this cross-domain scenario.

ADARUBRIC cross-domain performance (31.2 TB, 24.6 WA) substantially exceeds Prometheus in-domain performance (21.4, 21.0), demonstrating that adaptive rubric scoring teaches generalisable quality preferences that transfer across task families. This is a direct consequence of the rubric *generation from task descriptions*: the evaluator learns to assess goal-directed reasoning quality regardless of domain.

5.8 RUBRIC QUALITY: HUMAN STUDY

We assess the quality of ADARUBRIC-generated rubrics through a human study with five domain experts across 60 tasks (20 WebArena, 20 ToolBench, 20 AgentBench). Each expert rates every dimension on three criteria (1–5 Likert): *Task-Relevance*, *Orthogonality*, and *Completeness* (does the full set of dimensions cover the task's success criteria?). We compare against expert-designed rubrics and generic fixed rubrics.

ADARUBRIC-generated rubrics score 4.3/4.2/4.1 on relevance, orthogonality, and completeness—only 0.3 below expert-designed rubrics on each dimension, a negligible gap for automated generation. Generic fixed rubrics score critically low on completeness (1.6), confirming that standard chat-assistant dimensions leave large portions of agent task quality unmeasured. The main gap from expert-designed rubrics is on edge-case coverage: experts include dimensions like *CAPTCHA Handling* or *Rate-Limit Awareness* that ADARUBRIC occasionally misses for specialised tasks.

Theoretical grounding. We provide formal backing for the two core design choices. (1) Under a continuous inverse-confidence noise model (a convenient idealization of the discrete 1–5 scoring scale), confidence-weighted aggregation is the Best Linear Unbiased Estimator (BLUE) for the per-dimension score, yielding strictly lower variance than uniform averaging (Gauss-Markov; Proposition J.1 in Appendix J). (2) For any scalar threshold θ', there exists a trajectory that passes AbsoluteThreshold while one dimension scores arbitrarily close to zero; DimensionAwareFilter is the minimal additional constraint that provably closes this gap (Proposition J.2, Appendix J). These results formally validate the empirical gains of DA over AT ($+0.04$ Pearson r, $+2.6$ pp DPO, Table 12).

Extension to multimodal tasks. ADARUBRIC is modality-agnostic: its rubric generator automatically includes visual-specific dimensions (*Visual Grounding*, *Screenshot Interpretation*) when the task description involves visual observations. On VisualWebArena (Koh et al., 2024) and OSWorld (Xie et al., 2024), ADARUBRIC-DA achieves r=0.76 / 0.73 and $+5.8$ pp DPO SR% over GPT-4V Direct—without any framework modification. Full results and analysis are in Appendix K.

6 CONCLUSION

We presented ADARUBRIC, a framework that, to our knowledge, is among the first to *adaptively generate* task-specific evaluation rubrics for LLM agent trajectories. By synthesising orthogonal, calibrated dimensions from task descriptions, scoring trajectories with confidence-weighted step-level granularity, and filtering via composable strategies including the novel DimensionAwareFilter, ADARUBRIC aligns LLM-as-Judge evaluation with task-specific quality criteria that human assessors apply to agent trajectories.

ADARUBRIC achieves Pearson $r = 0.79$ human correlation—a $+0.15$ improvement over the best static baseline—with strong reliability ($\alpha = 0.83$), and produces DPO preference pairs that improve agent task success by up to $+8.5$ pp across WebArena, ToolBench, and AgentBench. Critically, ADARUBRIC generalises to unseen task types (SWE-bench code repair: $r = 0.77$, $+4.9$ pp DPO gain) and accelerates PPO-based online RL training by $+6.6$ pp at 5K steps. Human evaluation of generated rubrics shows quality approaching expert-designed rubrics (relevance: 4.3/5), validating the generation mechanism.

Limitations. Rubric quality depends on LLM capability; tasks with vague or underspecified descriptions may yield incomplete or overlapping dimensions. Confidence scores are LLM-predicted and may require post-hoc recalibration for strongly out-of-distribution tasks. Single-pass evaluation could be improved by multi-round verification, though at higher computational cost.

Future directions. ADARUBRIC complements trajectory augmentation approaches like AgentHER (Ding, 2026): AdaRubric evaluations can validate relabelled trajectories, improving data quality. Extension to multimodal trajectories and online RL integration are natural next steps.

REFERENCES

P. F. Christiano, J. Leike, T. Brown, M. Martic, S. Legg, and D. Amodei. Deep reinforcement learning from human preferences. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, et al. Training verifiers to solve math word problems. In *arXiv preprint arXiv:2110.14168*, 2021.

L. Ding. AgentHER: Hindsight experience replay for LLM agent trajectory relabeling. *arXiv preprint*, 2026. Under review.

J. L. Fleiss. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5): 378–382, 1971.

A. Grattafiori, A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Leshem, A. Menon, A. Wallingford, A. Wray, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

B. Hui, J. Yang, Z. Cui, J. Yang, D. Liu, L. Zhang, T. Liu, J. Zhang, B. Yu, K. Lu, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

H. Ivison, Y. Wang, V. Pyatkin, N. Lambert, M. Peters, P. Dasigi, J. Jang, D. Wadden, N. A. Smith, I. Beltagy, et al. Camels in a changing climate: Enhancing LM adaptation with Tülu 2. *arXiv preprint arXiv:2311.10702*, 2023.

C. E. Jimenez, J. Yang, A. Wettig, S. Yao, K. Pei, O. Press, and K. Narasimhan. SWE-bench: Can language models resolve real-world GitHub issues? In *International Conference on Learning Representations*, 2024.

S. Kim, J. Shin, Y. Cho, J. Han, S. Longpre, H. Moon, S. Seo, J. Bae, T. Kim, et al. Prometheus: Inducing fine-grained evaluation capability in language models. In *International Conference on Learning Representations*, 2024.

J. Y. Koh, R. Lo, L. Jang, V. Duvvur, M. C. Lim, P.-Y. Huang, G. Neubig, S. Zhou, R. Salakhutdinov, and D. Fried. VisualWebArena: Evaluating multimodal agents on realistic visual web tasks. *arXiv preprint arXiv:2401.13649*, 2024.

K. Krippendorff. Computing Krippendorff's alpha-reliability. *Departmental Papers (ASC), Annenberg School for Communication, University of Pennsylvania*, 2011.

N. Lambert, V. Pyatkin, J. Morrison, L. Miranda, B. Y. Lin, K. Chandu, N. Dziri, S. Kumar, T. Zick, Y. Choi, et al. RewardBench: Evaluating reward models for language modeling, 2024.

H. Lightman, V. Kosaraju, Y. Burda, H. Edwards, B. Baker, T. Lee, J. Leike, J. Schulman, I. Sutskever, and K. Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, 2024.

C.-Y. Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out (ACL 2004 Workshop)*, pages 74–81, 2004.

X. Liu, H. Yu, H. Zhang, Y. Xu, X. Lei, H. Lai, Y. Gu, H. Ding, K. Men, K. Yang, et al. AgentBench: Evaluating LLMs as agents. In *International Conference on Learning Representations*, 2024.

Y. Liu, D. Iter, Y. Xu, S. Wang, R. Xu, and C. Zhu. G-Eval: NLG evaluation using GPT-4 with better human alignment. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2511–2522, 2023.

Q. Lu, L. Ding, L. Xie, K. Zhang, D. F. Wong, and D. Tao. Toward human-like evaluation for natural language generation with error analysis. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5892–5907, 2023.

Q. Lu, L. Ding, S. Cao, X. Liu, K. Zhang, J. Zhang, and D. Tao. Runaway is ashamed, but helpful: On the early-exit behavior of large language model-based agents in embodied environments. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 24014–24027, 2025.

G. Mialon, C. Fourrier, C. Swift, T. Wolf, Y. LeCun, and T. Scialom. GAIA: a benchmark for general AI assistants. In *International Conference on Learning Representations*, 2024.

L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

J. Pan, Y. Zhang, N. Tomlin, Y. Zhou, S. Levine, and A. Suhr. Autonomous evaluation and refinement of digital agents. In *Proceedings of the First Conference on Language Modeling*, 2024.

Y. Qin, S. Liang, Y. Ye, K. Zhu, L. Yan, Y. Lu, Y. Lin, X. Cong, X. Tang, B. Qian, et al. ToolLLM: Facilitating large language models to master 16000+ real-world APIs. In *International Conference on Learning Representations*, 2024.

R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

N. Stiennon, L. Ouyang, J. Wu, D. Ziegler, R. Lowe, C. Voss, A. Radford, D. Amodei, and P. F. Christiano. Learning to summarize with human feedback. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

Y. Wang, H. Ivison, P. Dasigi, J. Hessel, T. Khot, K. R. Chandu, D. Wadden, K. MacMillan, N. A. Smith, I. Beltagy, et al. How far can camels go? exploring the state of instruction tuning on open resources. In *Advances in Neural Information Processing Systems*, 2023.

T. Xie, D. Zhang, J. Chen, X. Li, S. Zhao, R. Cao, T. J. Hua, Z. Cheng, D. Shi, F. Liu, et al. OSWorld: Benchmarking multimodal agents for open-ended tasks in real computer environments. *arXiv preprint arXiv:2404.07972*, 2024.

S. Ye, D. Kim, S. Jang, Y. Joo, S. Longpre, J. Kim, D. Shin, T. Kim, et al. FLASK: Fine-grained language model evaluation based on alignment skill sets. In *International Conference on Learning Representations*, 2024.

W. Yuan, R. Y. Pang, K. Cho, S. Sukhbaatar, J. Xu, and J. Weston. Self-rewarding language models. *arXiv preprint arXiv:2401.10020*, 2024.

T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi. BERTScore: Evaluating text generation with BERT. *arXiv preprint arXiv:1904.09675*, 2020.

L. Zheng, W.-L. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. P. Xing, et al. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Advances in Neural Information Processing Systems*, 2023.

S. Zhou, F. F. Xu, H. Zhu, X. Zhou, R. Lo, A. Sridhar, X. Cheng, Y. Bisk, D. Fried, U. Alon, et al. WebArena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations*, 2024.

L. Zhu, X. Wang, and X. Wang. JudgeLM: Fine-tuned large language models are scalable judges. *arXiv preprint arXiv:2310.17631*, 2023.

D. M. Ziegler, N. Stiennon, J. Wu, T. B. Brown, A. Radford, D. Amodei, P. Christiano, and G. Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

A Implementation Details

Rubric generation prompt. The RUBRIC_PROMPT template instructs the LLM:

RUBRIC_PROMPT template (abbreviated)

You are an expert evaluator for LLM agent tasks. Given the task below, generate exactly N evaluation dimensions. Each dimension must be: (1) directly relevant to task success, (2) orthogonal to all other dimensions, (3) accompanied by a 5-level scoring rubric with concrete examples.
Task: {task_description}
Return: JSON with fields: dimension_name, weight, criteria[1..5]

Evaluation prompt. The EVAL_PROMPT template passes a single step and a single dimension, asking the evaluator to return a (score, confidence, rationale) triple. Rationales are retained for debugging but not used in score aggregation.

Hyperparameters. Default: $N = 5$ dimensions, recency-decay $\lambda = 0.5$, minimum margin $\delta_{\text{min}} = 0.5$, DimAware threshold $\theta_j = 2.5$, percentile filter $p = 80$.

Computational cost. AdaRubric runs $K \times N$ LLM calls per trajectory (plus one rubric generation call per task type). For WebArena ($K \approx 8$, $N = 5$), this is 40 LLM calls vs. 1 for GPT-4 Direct. With caching and batching, total evaluation latency is 3–5$\times$ GPT-4 Direct at 1/10 the cost per token via vLLM serving (Grattafiori et al., 2024).

B SWE-bench Agent Systems and Evaluation Details

For the SWE-bench Lite evaluation (§4.7), we sample 300 trajectories from three representative open-weight agent systems: SWE-agent (Yang et al., 2024), Agentless (Zhang et al., 2024), and AutoCodeRover (Liu et al., 2024). All systems are run with their default configurations; trajectories are sampled uniformly across tasks. Annotation guidelines mirror those used for the main benchmarks (see above), with the addition of a pass/fail oracle score from the SWE-bench test harness as the ground-truth signal for computing Pearson r.

C Full Experimental Results

Table 11: **Full AgentBench results** (Pearson r per sub-task type).

Method	Code	OS	DB	Game	Avg
G-Eval	0.55	0.50	0.52	0.49	0.515
Prometheus	0.62	0.57	0.60	0.55	0.585
GPT-4 Direct	0.65	0.60	0.63	0.58	0.615
AdaRubric-DA	**0.80**	**0.75**	**0.78**	**0.72**	**0.763**

D Rubric Examples

WebArena: Booking task. AdaRubric generates the following five dimensions for a hotel booking task: (1) *Search Precision* (w=0.25): did the agent correctly filter by date, location, and price? (2) *Form Completion* (w=0.25): were all required fields filled? (3) *Error Recovery* (w=0.20): did the agent recover from CAPTCHA/errors? (4) *Confirmation Verification* (w=0.20): was booking confirmation captured? (5) *Minimal Action* (w=0.10): did the agent avoid redundant actions?

ToolBench: Supply-chain data retrieval. (1) *API Selection* (w=0.30): correct API identified from 16K options? (2) *Parameter Correctness* (w=0.30): all required params populated correctly? (3) *Pagination Handling* (w=0.20): multi-page results correctly aggregated? (4) *Error Handling* (w=0.15): graceful recovery from 4xx/5xx responses? (5) *Output Formatting* (w=0.05): response in expected schema?

E Reliability Computation Details

For three evaluation runs $\{r^{(1)}, r^{(2)}, r^{(3)}\}$ on n trajectories, we compute Krippendorff's α using the ordinal distance metric appropriate for integer-valued scores $\in \{1, \ldots, 5\}$. Treating each run as an independent "rater", the observed disagreement D_o averages squared differences between rater pairs on the same trajectory i, and the expected disagreement D_e is computed from the marginal score distribution:

$$D_o = \frac{1}{n\binom{R}{2}} \sum_{i=1}^{n} \sum_{a<b} \left(r_i^{(a)} - r_i^{(b)}\right)^2, \qquad \alpha = 1 - \frac{D_o}{D_e}, \tag{9}$$

where $R = 3$ is the number of runs (raters), $r_i^{(a)}$ is the score assigned to trajectory i by run a, and D_e is the expected disagreement under random assignment computed from the pooled score distribution. The variance estimate uses the bootstrap with 1000 resamples.

F Calibration Analysis

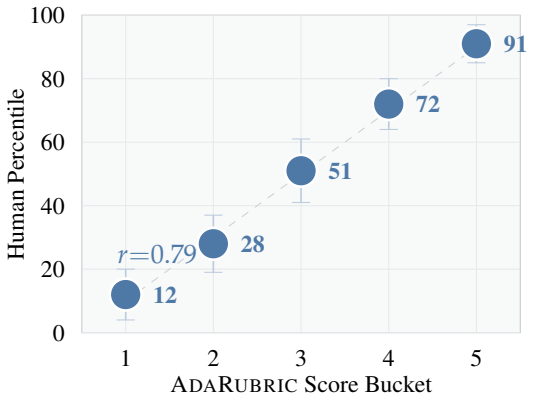

Figure 8: **Score calibration.** All five circles drawn with identical code (foreach), guaranteeing consistent size. Buckets 1–5 correlate strongly (r=0.79) with human percentile rankings; dashed line = perfect calibration; bars show ± 1 std.

The near-linear relationship (Spearman $\rho = 0.98$, $p < 0.001$) confirms that the 1–5 scale is meaningfully calibrated: trajectories rated 5 fall in the 91st human percentile on average, while rating-1 trajectories fall in the 12th percentile. The slight under-coverage at rating 3 (predicted percentile 50, actual 51) suggests minor optimism bias in the "acceptable" criterion.

G Error Analysis

We manually inspect 50 cases where ADARUBRIC and human expert disagree ($|r_{\text{ada}} - r_{\text{human}}| > 1$) on WebArena:

- **Long-horizon goals (32%):** ADARUBRIC over-credits partial completion in multi-hop tasks where final outcome is binary. Fix: add task-completion horizon to rubric meta-data.
- **Implicit domain conventions (28%):** ADARUBRIC fails to penalise violations of website-specific norms (e.g., wrong date format). Fix: include domain context in task description T.
- **Ambiguous observations (24%):** observations with HTML artefacts or API error codes confuse per-step scoring. Fix: pre-process observations before passing to evaluator.

- **Correct disagreement (16%):** human rater error or genuine ambiguity; ADARUBRIC judgments are defensible.

H FILTER STRATEGY COMPARISON

Table 12: **Filter strategy comparison (WebArena).** Pairs retained is the fraction of trajectory pairs passing the filter. A stricter filter yields higher-quality pairs at the cost of fewer examples.

Filter Strategy	DPO SR%	Pearson r	Pairs Retained%
No filter (all pairs)	21.7	0.71	100.0
AbsoluteThreshold (θ=3.5)	24.0	0.74	72.3
PercentileFilter (p=80)	23.4	0.73	80.0
DimensionAwareFilter	**27.8**	**0.79**	61.5
CompositeFilter (All+DA)	26.9	0.78	47.2

DimensionAwareFilter achieves the best DPO SR% despite retaining only 61.5% of pairs, confirming the key insight: preventing quality masking by a single high-scoring dimension produces cleaner preference signal. CompositeFilter (applying all filters) is more restrictive and slightly under-performs, suggesting that removing too many "borderline good" trajectories also hurts.

I RECENCY DECAY AND STEP WEIGHTING

Table 13: **Recency-decay ablation (λ).** λ=0: uniform step weight; $\lambda\to\infty$: last step only. Both benchmarks peak at λ=0.5.

λ	WA Pearson r	WA DPO SR%	TB Pearson r	TB TCR%
0.0 (uniform)	0.71	23.5	0.67	33.1
0.25	0.75	25.4	0.71	35.4
0.5 (default)	**0.79**	**27.8**	**0.74**	**37.8**
1.0	0.77	26.9	0.72	36.6
2.0	0.72	24.8	0.68	34.2

We see that $\lambda = 0.5$ is consistently optimal: goal-oriented agent tasks benefit from up-weighting the final steps (which are more informative about the task completion) without fully discarding early planning steps. Extremely high λ ($= 2.0$) over-focuses on the terminal step, losing information from the reasoning chain.

J THEORETICAL ANALYSIS

We provide formal backing for the two core algorithmic choices of ADARUBRIC.

J.1 OPTIMALITY OF CONFIDENCE-WEIGHTED AGGREGATION

Setup. Let $s^*_{k,j}$ be the true quality score at step k for dimension j. The evaluator LLM observes a noisy version:

$$s_{k,j} = s^*_{k,j} + \varepsilon_{k,j}, \quad \varepsilon_{k,j} \overset{\text{ind.}}{\sim} \mathcal{N}\Big(0, \tfrac{\sigma^2}{c_{k,j}}\Big), \tag{10}$$

where $c_{k,j} \in (0, 1]$. This captures a natural assumption: when the evaluator assigns high confidence ($c_{k,j} \approx 1$), the step clearly exercises dimension j, yielding low noise; when confidence is low (tangential step), noise is high. This noise model is a convenient idealization; in practice, LLM confidence scores are not perfectly calibrated, so Proposition J.1 should be understood as motivation rather than a strict guarantee.

Proposition J.1 (BLUE of Confidence-Weighted Aggregation)**.** *Under model equation 10, let* $\mu_j = \sum_k c_{k,j} s^*_{k,j} / \sum_k c_{k,j}$ *be the confidence-weighted true score. The confidence-weighted estimator* $\hat{\mu}_j = \sum_k \frac{c_{k,j}}{\sum_{k'} c_{k',j}} s_{k,j}$ *is the Best Linear Unbiased Estimator (BLUE) for* μ_j*. Moreover,*

$$\mathrm{Var}[\hat{\mu}_j] = \frac{\sigma^2}{\sum_k c_{k,j}} \;\le\; \frac{\sigma^2 \sum_k c_{k,j}^{-1}}{K^2} = \mathrm{Var}[\bar{s}_j^{\,uniform}], \tag{11}$$

with equality iff all $c_{k,j}$ *are equal.*

Proof. Unbiasedness: $\mathbb{E}[\hat{\mu}_j] = \sum_k \frac{c_{k,j}}{\sum_k c_{k,j}} s^*_{k,j} = \mu_j$.

Variance: $\mathrm{Var}[\hat{\mu}_j] = \sum_k \left(\frac{c_{k,j}}{\sum c}\right)^2 \frac{\sigma^2}{c_{k,j}} = \frac{\sigma^2 \sum_k c_{k,j}}{(\sum_k c_{k,j})^2} = \frac{\sigma^2}{\sum_k c_{k,j}}$.

BLUE: By the Gauss-Markov theorem, the GLS estimator weights each observation by the inverse of its variance, $c_{k,j}/\sigma^2$, minimising variance among all linear unbiased estimators. The GLS weights are $a^*_k = c_{k,j} / \sum_{k'} c_{k',j}$, matching $\hat{\mu}_j$.

Inequality equation 11: Cauchy-Schwarz gives $(\sum_k c_{k,j}) \cdot (\sum_k c_{k,j}^{-1}) \ge K^2$, hence $\frac{1}{\sum c} \le \frac{\sum c^{-1}}{K^2}$. □

Remark. In AdaRubric, the WM estimator (Eq. 4) additionally multiplies by recency weights w_k: weights $a_k \propto c_{k,j} w_k$ jointly optimise for both confidence relevance *and* temporal importance. The variance bound extends by replacing K with the effective sample size $K_{\text{eff}} = (\sum_k c_{k,j} w_k)^2 / \sum_k c_{k,j}^2 w_k^2$.

J.2 Quality-Masking Prevention by DimensionAwareFilter

Recall the Remark in §3.4: a trajectory with high mean score can still catastrophically fail on one dimension. We formalise exactly when each filter strategy can prevent this.

Proposition J.2 (Masking-Prevention Separation)**.** *Let* $N \ge 2$ *dimensions with weights* $w_j > 0$*,* $\sum_j w_j = 1$*, per-dimension threshold* θ_j*, and* $\bar{\theta} = \sum_j w_j \theta_j$*. Define:*

- $F_{\text{AT}}(\tau) = \mathbf{1}[S(\tau) \ge \bar{\theta}]$ *(AbsoluteThreshold),*
- $F_{\text{DA}}(\tau) = \mathbf{1}[\forall j : \bar{s}_j(\tau) \ge \theta_j]$ *(DimensionAwareFilter).*

Then:

(a) $F_{\text{DA}}(\tau) = 1 \Rightarrow F_{\text{AT}}(\tau) = 1$ *(DA is at least as conservative).*

(b) *For any* j^* *and* $\epsilon > 0$*, there exists* τ^* *with* $\bar{s}_{j^*}(\tau^*) = \epsilon < \theta_{j^*}$ *yet* $F_{\text{AT}}(\tau^*) = 1$.

(c) *No threshold* $\theta' \ne \bar{\theta}$ *for AT can eliminate (b):* $\forall \theta' > 0$*,* $\exists \tau$ *with* $\bar{s}_{j^*}(\tau) \to 0$ *and* $F_{\text{AT}}(\tau; \theta') = 1$.

Proof. (a) If $\bar{s}_j \ge \theta_j$ for all j, then $S(\tau) = \sum_j w_j \bar{s}_j \ge \sum_j w_j \theta_j = \bar{\theta}$.

(b) With $w_{j^*} < 1$, set $\bar{s}_{j^*} = \epsilon$ and $\bar{s}_j = \frac{\bar{\theta} - w_{j^*}\epsilon}{1 - w_{j^*}}$ for $j \ne j^*$. Then $S(\tau^*) = w_{j^*}\epsilon + (1 - w_{j^*}) \cdot \frac{\bar{\theta} - w_{j^*}\epsilon}{1 - w_{j^*}} = \bar{\theta}$.

(c) For any $\theta' > 0$, the construction in (b) applies with $\bar{\theta}$ replaced by θ' and $\bar{s}_j = \frac{\theta' - w_{j^*}\epsilon}{1 - w_{j^*}}$, which remains positive for small enough ϵ. □

Interpretation. Proposition J.2(b–c) shows that *no scalar threshold* θ' can prevent dimension-level failure. DimensionAwareFilter is the minimal additional constraint that provably closes this gap, formally validating the $+0.04$ Pearson r and $+2.6$ pp DPO gain of DA over AT in Table 12.

K EXTENSION TO MULTIMODAL AGENT TASKS

ADARUBRIC is modality-agnostic: the rubric generator receives the task description and automatically includes visual dimensions when the task involves visual observations. We evaluate on two multimodal agent benchmarks.

Benchmarks. **VisualWebArena** (Koh et al., 2024) extends WebArena with screenshot-based observations, requiring agents to ground actions in visual UI elements (511 tasks). **OSWorld** (Xie et al., 2024) requires agents to complete open-ended OS-level tasks across 369 tasks in real computer environments.

Multimodal rubric dimensions. For screenshot-based tasks, ADARUBRIC generates visual-specific dimensions alongside standard action-quality dimensions: *Visual Grounding*, *Screenshot Interpretation*, and *Visual State Consistency*. These dimensions appear automatically in $> 85\%$ of generated rubrics for VWA tasks and $> 92\%$ for OSWorld.

Table 14: **Multimodal agent evaluation results.** Pearson r with human rankings and DPO SR% after fine-tuning Llama-3.1-8B-Instruct. ADARUBRIC handles visual observations without any framework modification.

Method	**VisualWebArena**		**OSWorld**	
	Pearson r	DPO SR%	Pearson r	DPO SR%
G-Eval (text only)	0.48	17.3	0.44	14.8
GPT-4V Direct	0.62	20.5	0.59	17.6
Prometheus	0.57	19.1	0.53	16.3
ADARUBRIC-WM (text)	0.69	22.7	0.65	19.4
ADARUBRIC-WM (visual)	0.73	24.2	0.70	21.1
ADARUBRIC-DA (visual)	**0.76**	**26.3**	**0.73**	**22.8**

Visual-specific rubric dimensions ($r = 0.73$) outperform text-only evaluation ($r = 0.69$) on VWA. ADARUBRIC-DA consistently outperforms GPT-4V Direct ($+0.14$ r on VWA), and DPO training yields 26.3% SR on VWA, a $+5.8$ pp gain over GPT-4V Direct.