Auto-Rubric as Reward: From Implicit Preferences to Explicit Generative Criteria

Juanxi Tian[1] **Fengyuan Liu**[1] **Jiaming Han**[3] **Yilei Jiang**[3]
Yongliang Wu[4] **Yesheng Liu**[1] **Haodong Li**[1] **Furong Xu**[2]
Wanhua Li[1,†]

[1]Nanyang Technological University [2]Ant Group
[3]MMLab, The Chinese University of Hong Kong [4]UIUC

Abstract

Aligning multimodal generative models with human preferences demands reward signals that respect the compositional, multi-dimensional structure of human judgment. Prevailing RLHF approaches reduce this structure to scalar or pairwise labels, collapsing nuanced preferences into opaque parametric proxies and exposing vulnerabilities to reward hacking. While recent Rubrics-as-Reward (RaR) methods attempt to recover this structure through explicit criteria, generating rubrics that are simultaneously reliable, scalable, and data-efficient remains an open problem. We introduce Auto-Rubric as Reward (ARR), a framework that reframes reward modeling from implicit weight optimization to explicit, criteria-based decomposition. Before any pairwise comparison, ARR externalizes a VLM's internalized preference knowledge as prompt-specific rubrics, translating holistic intent into independently verifiable quality dimensions. This conversion of implicit preference structure into inspectable, interpretable constraints substantially suppresses evaluation biases including positional bias, enabling both zero-shot deployment and few-shot conditioning on minimal supervision. To extend these gains into generative training, we propose Rubric Policy Optimization (RPO), which distills ARR's structured multi-dimensional evaluation into a robust binary reward, replacing opaque scalar regression with rubric-conditioned preference decisions that stabilize policy gradients. On text-to-image generation and image editing benchmarks, ARR-RPO outperforms pairwise reward models and VLM judges, demonstrating that explicitly externalizing implicit preference knowledge into structured rubrics achieves more reliable, data-efficient multimodal alignment, revealing that the bottleneck is the absence of a factorized interface, not a deficit of knowledge. Code is publicly available at `https://github.com/OpenEnvision/AutoRubric-as-Reward`.

1 Introduction

Human preferences are not arbitrary signals but structured, multidimensional judgments encompassing aesthetic value, semantic fidelity, and contextual appropriateness [20, 28, 47]. Aligning generative multimodal models with such preferences therefore demands more than calibration: it requires models to internalize and operationalize the explicit criteria that underpin human evaluation. Prevailing RLHF paradigms contravene this requirement. By collapsing composite preference structures into scalar scores [28, 47] or pairwise labels [20], they encode rich human judgment into opaque, entangled representations, discarding the very dimensions that confer interpretability and stability, and exposing the learning process to reward hacking [4, 11].

[†]Corresponding author: `wanhua.li@ntu.edu.sg`

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

Despite their extensive world knowledge and perceptual capabilities, contemporary VLMs exhibit systematic unreliability in modeling human preferences [17, 35]. Pointwise scoring reduces evaluation to a single scalar, providing no constraint on how improvement is achieved and allowing degenerate optimization strategies. Pairwise comparison, while more balanced, still operates on a latent decision boundary, leading to persistent positional biases that resist standard mitigations such as positional labeling or chain-of-thought prompting [25, 35]. Recent Rubrics as Reward (RaR) approaches attempt to recover structure through explicit criteria; however, their reliance on fixed or supervised rubric construction limits scalability, prompt specificity, and data efficiency, with these limitations becoming more pronounced when extended to multimodal generation settings.

The reframing recasts multimodal alignment as a representation problem: the bottleneck is not a deficit of preference knowledge, but the absence of a stable, factorized interface for applying it. Building on training-free rubric extraction from preference pairs [46], we propose **Auto-Rubric as Reward (ARR)**. ARR synthesizes instance-conditioned rubrics through a generate-verify-refine pipeline that induces discriminative criteria grounded in observable evidence, producing a compact set of verifiable, decision-relevant constraints spanning semantic fidelity, spatial consistency, compositional aesthetics, and edit faithfulness [12, 21, 32, 51]. These criteria compose a structured evaluation protocol for criterion-level comparison, supplanting holistic scoring. Unlike handcrafted rubrics or learned scalar rewards, ARR derives prompt-specific decision structures from minimal preference data with *no parameter updates*, yielding a highly data-efficient and interpretable interface. By externalizing preference structure into explicit, verifiable criteria, ARR replaces unstable latent comparisons with grounded discrimination, helping to reduce positional bias and mitigating reward hacking. Crucially, rubric quality scales with the underlying VLM's alignment with human preferences: stronger judges produce more precise criteria without additional supervision.

This formulation extends from evaluation to optimization. If preference is inherently factorized, reward should preserve that structure rather than collapse it. We therefore introduce **Rubric Policy Optimization (RPO)**, which uses ARR-generated criteria to produce binary preference decisions for policy optimization. Unlike prior rubric-based methods that apply criteria as auxiliary filters, RPO integrates rubric-conditioned judgments directly into the optimization objective, aligning gradient updates with interpretable dimensions of quality. This eliminates a separate reward model and mitigates reward hacking by grounding supervision in explicit criteria rather than learned proxies [4, 26]. Evaluation and generation are unified through a shared preference representation, where better understanding of human preferences in evaluation directly strengthens generative alignment.

Empirically, ARR improves preference accuracy over trained reward models and direct VLM judges by 1.7 to 6.3 points, while reducing positional bias and retaining strong zero-shot and few-shot generalization. When used for training, ARR-RPO yields further gains on text-to-image generation and image editing benchmarks [12, 16, 17, 24, 28, 37, 40, 43, 49] (e.g., GenEval: 0.66 to 0.80; DPG-Bench: 83.84 to 85.76). These improvements require no judge fine-tuning or large-scale reward annotation. The core insight is that the bottleneck in multimodal alignment lies not in acquiring more preference knowledge, but in providing a stable, factorized interface to apply it, precisely what explicit rubrics supply.

Our key contributions can be summarized as follows:

- **Auto-Rubric as Reward (ARR).** We propose a training-free framework that externalizes implicit human preferences into instance-conditioned, interpretable rubrics. It enables scalable multimodal evaluation with extremely high data efficiency, requiring only a few annotated samples.

- **Rubric Policy Optimization (RPO).** We introduce RPO, a policy optimization framework for contrastive preference learning. By conditioning on ARR-derived rubrics, RPO replaces scalar reward signals with structured, criterion-grounded comparisons.

- **Diagnosing the Interface Bottleneck.** Ablations reveal the core bottleneck is a missing factorized interface, not a knowledge deficit. ARR-RPO resolves this via explicit rubrics; cross-model and cardinality analyses confirm that deeper comprehension of intrinsic criteria, rather than scale or data volume, drives both evaluation robustness and generative improvement.

2 Related work

Multimodal reward modeling. RLHF underpins alignment across text-to-image generation, editing, and video synthesis. Early reward models such as PickScore, ImageReward, and HPS compress rich human preferences into scalar signals [20, 28, 47]. While effective for coarse ranking, such compression obscures preference structure and is prone to reward hacking and overfitting [4, 53]. Direct optimization methods eliminate explicit reward modeling but still rely on scalar or pairwise objectives, inheriting similar limitations in expressivity and robustness [11, 34]. Recent VLM-as-a-judge approaches leverage stronger multimodal priors, yet exhibit persistent biases, such as positional and symmetry bias, that are difficult to eliminate through prompting alone [17, 25, 35, 52]. Taken together, these methods suggest that the core limitation is not a lack of preference knowledge, but the absence of a structured interface for expressing and applying it. We address this by externalizing implicit preferences into explicit, prompt-conditioned rubrics, enabling factorized and verifiable evaluation in place of opaque scalar scoring.

Rubric as reward. To overcome the limitations of scalar evaluation, recent work has explored rubric-based formulations that decompose judgments into interpretable criteria. In language tasks, analytic rubric frameworks [30, 48] and LLM-Rubric [14] show that criterion-level assessment yields more stable and calibrated signals than holistic scoring [1, 19, 29]. AutoRubric [46] extends this idea by distilling generalizable criteria from preference data, yet remains confined to text-only evaluation. In multimodal settings, AutoRubric-R1V [18] compiles consistent reasoning steps from successful trajectories into problem-specific rubrics for process-level supervision, but it is designed for vision-language reasoning, not generative policy optimization. Despite these advances, no prior method in multimodal generation adopts auto-generated rubrics as the reward for both evaluation and training[22, 52]. We address this gap by treating rubrics as the direct preference interface, instantiating them as explicit, prompt-conditioned criteria that govern evaluation and provide the reward signal for optimization. This reframes alignment from implicit scalar optimization to structured discrimination over verifiable criteria, yielding a more interpretable and robust reward.

3 Methodology

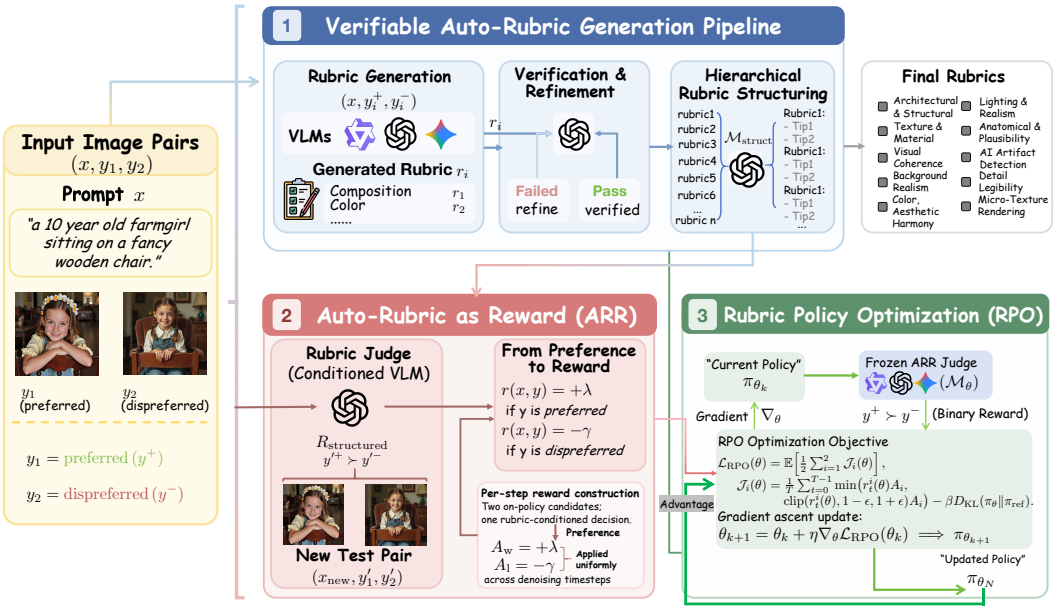


Figure 1: **Overview of the ARR-RPO framework.**(1) Rubrics are generated from preference pairs, verified, and hierarchically structured. (2) ARR converts rubric-conditioned VLM judgments into binary rewards. (3) RPO uses these rewards from a frozen judge to optimize the generative policy.

3.1 Problem formulation

We formulate preference learning as estimating the optimal parameters of a probabilistic model P_θ that, given a prompt x and candidate outputs y^+, y^-, assigns higher likelihood to the response better satisfying human intent. Preference alignment thus optimizes P_θ to capture and generalize human preferences, raising the central design question: how should the parameters θ be specified? We address this by decomposing the problem into ARR for evaluation and RPO for training (Figure 1).

Implicit preference modeling. For implicit preference modeling, given a pair of outputs (y^+, y^-) conditioned on the same input x, the human preference probability is typically defined using the Bradley-Terry (BT) model as follows:

$$P^*(y^+ \succ y^- \mid x) = \frac{\exp(r^*(x, y^+))}{\exp(r^*(x, y^+)) + \exp(r^*(x, y^-))} \tag{1}$$

where $*$ denotes the parameters corresponding to the true underlying human preference distribution. Here, r^* represents the ideal scalar reward model that perfectly reflects human preferences. In practice, since the true human preference distribution is inaccessible, we typically work with a pairwise preference dataset $\mathcal{D}$ that approximately captures human judgments. We can then parameterize a reward model r_ϕ and estimate the true parameters ϕ^* by solving the following optimization problem:

$$\mathcal{L}_R(r_\phi, \mathcal{D}) = -\mathbb{E}_{(x,y^+,y^-)\sim\mathcal{D}}\left[\log \sigma(r_\phi(x, y^+) - r_\phi(x, y^-))\right] \tag{2}$$

where σ is the logistic function.

Explicit preference modeling. In explicit preference modeling, we define the preference distribution by employing a VLM as a judge. Given a paired input (x, y^+, y^-), the LLM judge processes the prompt x along with the two candidate outputs and produces a binary preference decision that approximates the underlying human preference distribution P_θ:

$$P_\theta(y^+ \succ y^- | x) = \mathcal{M}_\theta(y^+ \succ y^- \mid x, y^+, y^-, R), \tag{3}$$

where R is a carefully pre-defined natural language rubric designed to enhance the VLM's ability to discern subtle differences in response quality. Here, $\mathcal{M}_\theta$ denotes the VLM enhanced by R, which serves as the judge and outputs a binary preference decision between the two candidates.

3.2 Auto-Rubric as Reward

Let $\mathcal{S}$ be the space of all possible rubrics. We aim to find the optimal rubric R^* that best approximates the underlying human preference distribution. Given an ideal preference model P^* instantiated by a highly capable LLM judge, the optimal rubric can be formulated as:

$$R^* = \arg\max_{R\subset\mathcal{S}} \sum_{i=1}^{N} \log P^*(y_i^+ \succ y_i^- | x_i, R) \tag{4}$$

Since the space of all possible rubric sets $\mathcal{S}$ is vast and discrete, directly optimizing the ideal objective is intractable. We therefore simplify the optimization target as selecting the best rubric subset:

$$R^* \approx \arg\max_{R\subset\mathcal{D}_R} \sum_{i=1}^{N} \mathbb{I}[\mathcal{M}_\theta(y_i^+ \succ y_i^- \mid x_i, y_i^+, y_i^-, R) = \text{correct}], \tag{5}$$

where $\mathcal{D}_R$ is a finite set of candidate rubrics. In the remainder of this section, we detail our approach for automatically constructing high-quality rubrics from data and demonstrate how these auto-generated rubrics can serve as an interpretable and effective reward signal when applied to reinforcement learning tasks.

Verifiable rubric generation. Given a pairwise preference dataset $\mathcal{D} = \{(x_i, y_i^+, y_i^-)\}_{i=1}^N$, we first generate K candidate rubrics per labeled pair offline, with $K = 5$ by default. For every pair (x_i, y_i^+, y_i^-), an VLM is prompted to produce a detailed natural language rubric r_i that explains why y_i^+ is preferred over y_i^-:

$$r_i = \mathcal{M}_{\text{gen}}(x_i, y_i^+, y_i^-). \tag{6}$$

To ensure quality, each generated rubric r_i is then verified by a separate judgment step. The verifier checks whether the rubric consistently supports the original preference:

$$v_i = \mathcal{M}_{\text{verify}}(x_i, y_i^+, y_i^-, r_i). \tag{7}$$

Because the verifier independently checks whether the generated rubric consistently recovers the original preference label, it acts as a weak safeguard against self-reinforcing errors: rubrics that fail this consistency test are refined or discarded, reducing the chance of amplifying idiosyncratic model biases that survive the initial generation step.

If verification fails ($v_i = \text{false}$), we iteratively refine the rubric up to a predefined maximum number of attempts $T_{\max}$:

$$r_i^{(t+1)} = \mathcal{M}_{\text{refine}}(x_i, y_i^+, y_i^-, r_i^{(t)}), \quad t = 0, 1, \ldots, T_{\max} - 1. \tag{8}$$

If the rubric still fails verification after $T_{\max}$ refinement attempts, it is discarded. After processing all K candidates for each pair in $\mathcal{D}$, we obtain a set of verified rubrics:

$$\mathcal{D}_R = \{r_{i,k} \mid v_{i,k} = \text{true}\}. \tag{9}$$

This verifiable generation process yields a high-quality, instance-specific rubric collection $\mathcal{D}_R$ directly grounded in the preference dataset.

Hierarchical rubric structuring. After verification, the rubric set $\mathcal{D}_R$ captures fine-grained, per-instance criteria but lacks the coherence required for consistent conditioning across arbitrary prompts. We therefore prompt an LLM to consolidate $\mathcal{D}_R$ into a single, hierarchically organized rubric. The LLM groups related criteria by semantic granularity and preference dimension, producing a compact evaluation protocol. The resulting structured rubric $R_{\text{structured}}$ is directly reused as a system-prompt component for the judge and as a reward conditioning signal during optimization, removing the need for per-instance rubric regeneration at deployment. Formally,

$$R_{\text{structured}} = \mathcal{M}_{\text{struct}}(\mathcal{D}_R), \tag{10}$$

where $\mathcal{M}_{\text{struct}}$ denotes the LLM prompted to perform hierarchical organization and prompt synthesis. See Appendix I for final rubric examples.

From rubric to reward. To successfully apply the auto-rubric method to reinforcement learning tasks, we need to convert the generated rubrics into a usable reward signal. Since the VLM judge produces binary preference decisions, we assign a positive constant reward to the preferred response y^+ and a negative constant reward to the dispreferred response y^-. Formally, given a prompt x and a pair of outputs (y^+, y^-), the reward for a candidate y is defined with respect to the other output y' as:

$$r(x, y; y') = \begin{cases} +\lambda & \text{if } \mathcal{M}_\theta(x, y, y', R) \text{ prefers } y, \\ -\gamma & \text{otherwise,} \end{cases} \tag{11}$$

where $\lambda, \gamma > 0$ are constant reward magnitudes and R denotes the learned rubric set.

3.3 Rubric Policy Optimization

Having established a mechanism for generating high-quality rubrics and converting them into verifiable reward signals, we now introduce **Rubric Policy Optimization (RPO)**, an online policy optimization algorithm that directly utilizes the rubric judge to guide the generative policy π_θ.

Unlike conventional RLHF and prior rubric-based methods in multimodal generation that reduce criteria to scalar composites or auxiliary filters, RPO directly leverages the VLM judge's binary preferences conditioned on explicit rubrics as the reward signal. For each generated sample, the

preferred output y^+ receives a positive constant reward $+\lambda$, while the dispreferred output y^- receives $-\gamma$. This yields a dense per-step training objective that preserves the advantages of rubric-based evaluation while remaining compatible with standard policy gradient methods.

The resulting RPO objective is defined as:

$$\mathcal{L}_{\mathrm{RPO}}(\theta) = \mathbb{E}_{h\sim\mathcal{D},\,\{x^i_{0:T}\}^2_{i=1}\sim\pi_\theta}\left[\frac{1}{2}\sum_{i=1}^{2}\left(\frac{1}{T}\sum_{t=0}^{T-1}\min\Big(r^i_t(\theta)A_i,\right.\right.$$
$$\left.\left.\mathrm{clip}(r^i_t(\theta),1-\epsilon,1+\epsilon)A_i\Big) - \beta\, D_{\mathrm{KL}}(\pi_\theta\|\pi_{\mathrm{ref}})\right)\right]. \tag{12}$$

where the importance ratio at each timestep is

$$r^i_t(\theta) = \frac{\pi_\theta(x^i_{t-1}\mid x^i_t, h)}{\pi_{\theta_{\mathrm{old}}}(x^i_{t-1}\mid x^i_t, h)}. \tag{13}$$

Per-step reward construction. For a given prompt h (which may include both text condition c and the current rubric R), we sample two trajectories from the current policy π_θ. The VLM judge, conditioned on the learned rubric, produces a binary preference decision between the two trajectories. The winning trajectory is assigned advantage $A_w = +\lambda$ and the losing one $A_l = -\gamma$. This per-trajectory advantage is then uniformly distributed across all denoising (or generation) timesteps, providing a dense training signal that directly reflects rubric-guided human preference.

Online optimization and robustness. RPO optimizes the policy online: each iteration samples prompts from $\mathcal{D}$, generates two candidates from π_θ, evaluates them in one call to the frozen rubric judge using the cached $R_{\text{structured}}$, and applies the gradient of $\mathcal{L}_{\mathrm{RPO}}(\theta)$. Because rewards come from a frozen VLM judge conditioned on explicit rubrics rather than a trainable scalar model, RPO helps mitigate reward hacking. As the policy evolves, the judge continues to apply the same task rubric. PPO-style clipping and KL regularization further stabilize training and enable exploration aligned with the multi-dimensional criteria in the rubrics.

4 Experiments

We evaluate ARR as a preference evaluator and as a structured reward for generative policy optimization. Experiments on multimodal understanding, text-to-image generation, and image editing benchmarks compare against trained reward models and direct VLM judges to assess gains in evaluative reliability and downstream performance.

4.1 Experimental setup

Evaluation benchmarks. Evaluator fidelity is measured on three established testbeds: MM-RewardBench2 [17], which provides fine-grained diagnostic splits across multimodal reward scenarios; HPDv3 (test set) [28], a large-scale text-to-image preference corpus comprising 14,400 pairwise human judgments; and EditReward-Bench [43], specifically curated to probe instruction adherence in image editing. For generative quality assessment, we adopt GenEval [12], DPG-Bench [16], TIIF(test-mini-short)[40], and UniGenBench++[37] for text-to-image synthesis, complemented by GEdit-Bench[24] and ImgEdit[49] for editing tasks.

Baselines and implementation. For human preference evaluation, we compare against a suite of state-of-the-art trained reward models, including HPSv3 [28], PickScore [20], ImageReward [47], UnifiedReward[39] and UnifiedReward-Thinking [38], and EditReward [43], alongside representative VLM judges such as Qwen3-VL [2], GPT-5 [33], and Gemini 3.1 Pro [13].

Following the common practice in recent multimodal alignment and generation research [17, 22, 34], we adopt FLUX.1-dev [5] and Qwen-Image-Edit-2509 [41] as base models for image generation and editing, respectively. We perform post-training with RPO on LoRA-adapted versions of these models. Training prompts are drawn from ShareGPT-4o-Image [8]. Unless otherwise specified, ARR generates five candidate rubrics per labeled training pair offline and consolidates the verified candidates into one cached rubric per task and experimental configuration. We further contextualize results against leading contemporary generative models.

Table 1: **Evaluator performance across four preference benchmarks.** Accuracy (%) denotes agreement with human preference labels. The best result in each column is **bold**. Blue-shaded rows indicate ARR; green values indicate absolute gains over the corresponding baseline VLM judge.

Method	HPDv3 Acc.	MM-RewardBench2 (T2I) Acc.	MM-RewardBench2 (Edit) Acc.	EditReward-Bench Acc.
Trained Reward Model				
PickScore	65.6	58.6	—	—
ImageReward	58.6	54.0	—	—
UnifiedReward	66.0	59.8	—	—
UnifiedReward-Thinking	68.1	66.0	—	—
HPSv3	76.9	60.2	—	—
EditReward	—	—	67.2	56.45
VLM-as-Judge w/o ARR				
Qwen3-VL-8B	67.2	57.6	59.2	54.01
w/ ARR	70.2 ($\pm$ 0.2) $\uparrow$3.0	62.7 ($\pm$ 0.2) $\uparrow$5.1	65.5 ($\pm$ 0.3) $\uparrow$6.3	57.22 ($\pm$ 0.1) $\uparrow$3.21
GPT-5	72.4	70.5	73.8	57.53
w/ ARR	76.1 ($\pm$ 0.2) $\uparrow$3.7	74.7 ($\pm$ 0.4) $\uparrow$4.2	77.5 ($\pm$ 0.3) $\uparrow$3.7	61.01 ($\pm$ 0.1) $\uparrow$3.48
Gemini 3.1 Pro	76.6	75.1	77.4	61.23
w/ ARR	**78.3** ($\pm$ 0.1) $\uparrow$1.7	**78.9** ($\pm$ 0.2) $\uparrow$3.8	**79.2** ($\pm$ 0.2) $\uparrow$1.8	**63.27** ($\pm$ 0.2) $\uparrow$2.04

4.2 Human preference quality

We evaluate ARR as a preference evaluator on three standard benchmarks: HPDV3[28], which provides 1.17M human pairwise comparisons for text-to-image; MM-RewardBench2[17], with 4,000 expert-annotated preference pairs spanning four tasks; and EditReward-Bench, covering 13 subtasks of instruction-guided editing. For each benchmark, we report pairwise preference accuracy, defined as the fraction of test pairs where the model's predicted preference matches the human judgment. For each ARR configuration, we report the mean and standard deviation of the accuracies from three independent evaluation runs.

Results. Table 1 reports preference accuracy. Pairwise reward models specialize narrowly (e.g., HPSv3 drops from 76.9% on HPDv3 to 60.2% on MM-RewardBench2 T2I; EditReward falls from 67.2% to 56.5% on the broader EditReward-Bench), while direct VLM judges generalize better yet still struggle on challenging splits (Gemini 3.1 Pro: 75.1–77.4% on the first three columns but only 61.2% on EditReward-Bench). ARR conditioning consistently improves all judges by 1.7–6.3 points, with Gemini 3.1 Pro + ARR reaching state-of-the-art on all four benchmarks. Critically, base VLMs exhibit severe positional bias ($\Delta = 30.2$–34.6; Table 5); ARR reduces this gap to 27.8–31.6 (zero-shot) and to 8.9–10.3 with guidance. Gains persist across model families (Table 6), confirming that rubric quality, not generator-judge co-adaptation, drives results. Full results are in Table 10.

> **Takeaway:** Rubric conditioning does not merely improve accuracy; it reframes evaluation from preference matching to criteria-aligned verification. By externalizing standards *prior* to comparison, ARR mitigates latent biases in implicit preference models, yielding structured, task-adaptive decisions that directly support robust reward modeling.

4.3 Image generation and editing performance

We evaluate ARR-RPO on six benchmarks: GenEval [12], DPG-Bench [16], TIIF [40], and UniGen-Bench++ [37] for text-to-image generation; GEdit-Bench [24] and ImgEdit [49] for instruction-guided image editing. ARR-RPO fine-tunes FLUX.1.dev [5] and Qwen-Image-Edit-2509 [41] using ARR-generated rubrics as binary reward signals. We instantiate ARR with three VLMs, Qwen3-VL-8B [2], GPT-5 [33], and Gemini 3.1 Pro [13], to examine how rubric quality scales with judge capability.

Table 2: **Generative performance across T2I and image editing benchmarks.** Blue-shaded rows mark ARR-RPO variants; green values indicate absolute gains over the corresponding baseline.

Method	Text-to-Image					Image Editing	
	GenEval	DPG-Bench	TIIF	UniGenBench++		GEdit-Bench	ImgEdit
				Short	Long		
Specialist Model (T2I)							
Emu3	0.54	80.60	—	45.42	50.59	—	—
JanusFlow	0.63	79.68	—	47.10	54.80	—	—
FLUX.1-Dev	0.66	83.84	71.09	60.97	69.42	—	—
DALLE-3	0.67	83.50	74.96	68.85	70.82	—	—
Show-o2	0.76	86.14	—	61.90	70.33	—	—
OmniGen2	0.80	83.57	—	63.09	71.39	—	—
BAGEL	0.82	85.07	71.50	59.91	71.26	—	—
ARR-RPO / T2I (Ours)							
w/ RPO-Qwen3vl-8B-ARR	0.74 ↑0.08	85.03 ↑1.19	74.92 ↑3.83	64.17 ↑3.20	71.82 ↑2.40	—	—
w/ RPO-GPT-5-ARR	0.78 ↑0.12	85.41 ↑1.57	76.18 ↑5.09	65.36 ↑4.39	72.41 ↑2.99	—	—
w/ RPO-Gemini 3.1 Pro-ARR	0.80 ↑0.14	85.76 ↑1.92	76.85 ↑5.76	65.89 ↑4.92	72.93 ↑3.51	—	—
Specialist Model (Image Editing)							
Instruct-Pix2Pix	—	—	—	—	—	3.68	1.88
AnyEdit	—	—	—	—	—	3.21	2.45
Step1X-Edit	—	—	—	—	—	6.97	3.06
Qwen-Image-Edit-2509	—	—	—	—	—	7.54	4.35
UniWorld-Qwen-Image-Edit	—	—	—	—	—	7.76	4.48
ARR-RPO / Image Editing (Ours)							
w/ RPO-Qwen3vl-8B-ARR	—	—	—	—	—	7.66 ↑0.12	4.38 ↑0.03
w/ RPO-GPT-5-ARR	—	—	—	—	—	7.72 ↑0.18	4.40 ↑0.05
w/ RPO-Gemini 3.1 Pro-ARR	—	—	—	—	—	7.85 ↑0.31	4.43 ↑0.08

Results. Figure 2 and Table 2 report generative performance. Two patterns emerge. First, ARR-RPO consistently outperforms specialist baselines. For T2I, optimizing FLUX.1.dev with ARR rubrics lifts GenEval (0.66→0.80), DPG-Bench (83.84→85.76), TIIF (71.09→76.85), and UniGenBench++ Short (60.97→65.89). In editing, ARR-RPO elevates Qwen-Image-Edit-2509 on GEdit-Bench (7.54→7.85) and ImgEdit (4.35→4.43). Second, generated samples (Figure 3) exhibit marked improvements in visual quality and edit fidelity, aligning more closely with the multidimensional nature of human preferences. See Appendix 8 for full results.

> **Takeaway:** Rubrics externalize preference structure in a way that makes evaluation directly usable as reward. By grounding policy gradients in rubric-conditioned decisions rather than scalar proxies, ARR-RPO replaces opaque optimization with criteria-aligned improvement, enabling a direct transfer of evaluator fidelity into generative quality without reward model training or architectural change.

4.4 Ablation analysis

As shown in Figure 4(a), substantial positional bias ($\Delta = 30.2$ to 34.6) remains consistent across model scales in the absence of rubric conditioning. This suggests that the bias is not primarily due to insufficient model capacity, but is instead rooted in how preferences are implicitly encoded. Zero-shot ARR provides a modest reduction in bias (Δ decreases by 3.0 to 4.8), while human-guided rubrics lead to a much more pronounced improvement (Δ reduced to 8.9 to 10.3). These results indicate that making evaluation criteria explicit can significantly improve stability.

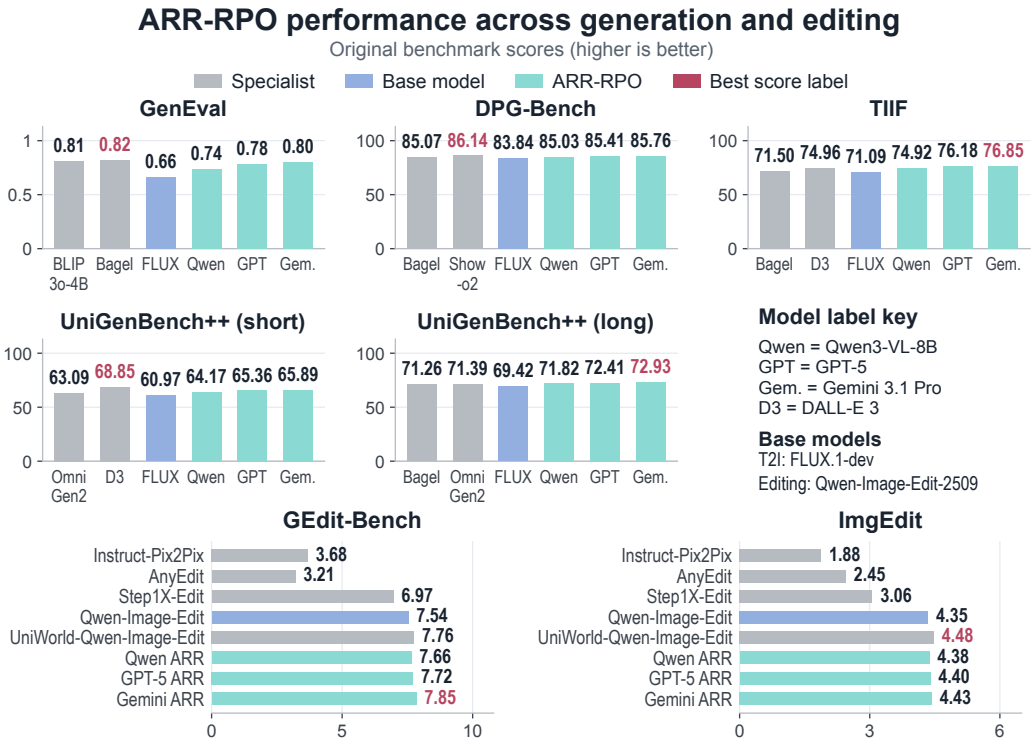


Figure 2: **Performance comparison of ARR-RPO variants against specialist models** across text-to-image generation (top) and image editing (bottom) benchmarks.

Figure 4(b) further shows that rubrics generalize across different model families (see Appendix C). Even when applied to weaker generators, transferred rubrics recover more than half of the performance gap compared to same-family settings. This observation suggests that the effectiveness of ARR is closely related to the quality and structure of the rubric itself, rather than reliance on tight coupling between the generator and the evaluator.

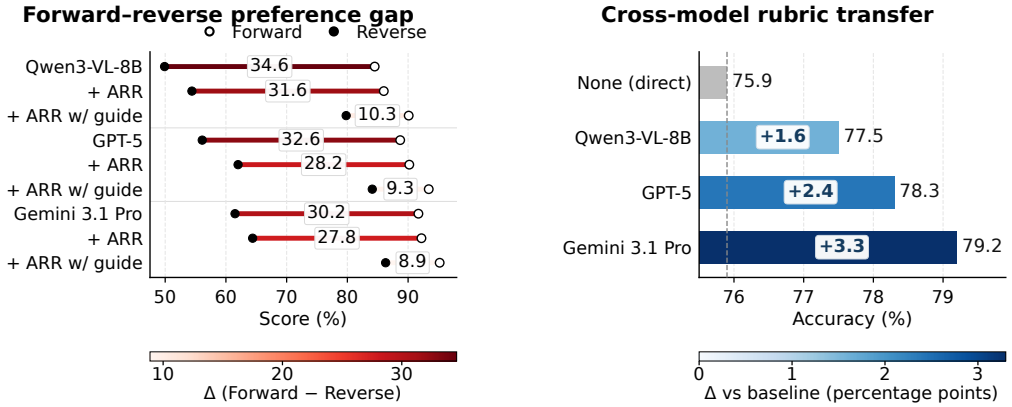
Furthermore, this interpretation is supported by the rubric cardinality ablation (Appendix D), where increasing rubric dimensionality consistently improves accuracy, indicating that ARR's gains arise from both finer-grained factorization of preference structure and the quality of the resulting rubric content, rather than model capacity or evaluator–generator coupling.

5 Conclusion

We present a unified ARR and RPO framework bridging multimodal preference evaluation and generative alignment. While prevailing approaches rely on implicit, entangled scalar signals that obscure underlying criteria and introduce systematic biases, ARR automatically generates instance-conditioned rubrics by prompting VLMs to externalize latent human preferences into explicit, interpretable criteria. These rubrics provide structured, factorized reward signals for RPO, enabling contrastive preference learning with fine-grained supervision across independent quality dimensions. Together, ARR and RPO replace opaque scalar rewards with explicit, composable criteria, consistently improving both evaluation reliability and generation quality without additional supervision or architectural modifications. This externalization of preference structure offers a principled, scalable pathway toward compositional alignment with nuanced, multidimensional human intent.



Figure 3: **Text-to-image and image editing examples** (ARR-RPO Gemini 3.1 Pro).



(a) (b)

Figure 4: **Ablation studies on ARR.** (a) Forward–reverse preference gaps across evaluators. (b) Cross-model rubric transfer with a fixed judge.

Acknowledgments and Disclosure of Funding

This research was supported in part by Nanyang Technological University through its Nanyang Assistant Professorship Startup Grant (025661-00012). The authors declare no competing interests.

References

[1] Zachary Ankner, Mansheej Paul, Brandon Cui, Jonathan D. Chang, and Prithviraj Ammanabrolu. Critique-out-Loud reward models. *arXiv preprint arXiv:2408.11791*, 2024.

[2] Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-VL technical report. *arXiv preprint arXiv:2511.21631*, 2025.

[3] James Betker, Gabriel Goh, Li Jing, Tim Brooks, Jianfeng Wang, Linjie Li, Long Ouyang, Juntang Zhuang, Joyce Lee, Yufei Guo, Wesam Manassra, Prafulla Dhariwal, Casey Chu, Yunxin Jiao, and Aditya Ramesh. Improving image generation with better captions. Technical report, OpenAI, 2023. `https://cdn.openai.com/papers/dall-e-3.pdf`.

[4] Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. In *The Twelfth International Conference on Learning Representations*, 2024.

[5] Black Forest Labs. FLUX.1 [dev] model card. Hugging Face, 2024.

[6] Tim Brooks, Aleksander Holynski, and Alexei A. Efros. InstructPix2Pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18392–18402, 2023.

[7] Jiuhai Chen, Zhiyang Xu, Xichen Pan, Yushi Hu, Can Qin, Tom Goldstein, Lifu Huang, Tianyi Zhou, Saining Xie, Silvio Savarese, et al. BLIP3-o: A family of fully open unified multimodal models—architecture, training and dataset. *arXiv preprint arXiv:2505.09568*, 2025.

[8] Junying Chen, Zhenyang Cai, Pengcheng Chen, Shunian Chen, Ke Ji, Xidong Wang, Yunjin Yang, and Benyou Wang. ShareGPT-4o-Image: Aligning multimodal models with GPT-4o-level image generation. *arXiv preprint arXiv:2506.18095*, 2025.

[9] Xiaokang Chen, Zhiyu Wu, Xingchao Liu, Zizheng Pan, Wen Liu, Zhenda Xie, Xingkai Yu, and Chong Ruan. Janus-Pro: Unified multimodal understanding and generation with data and model scaling. *arXiv preprint arXiv:2501.17811*, 2025.

[10] Chaorui Deng, Deyao Zhu, Kunchang Li, Chenhui Gou, Feng Li, Zeyu Wang, Shu Zhong, Weihao Yu, Xiaonan Nie, Ziang Song, Guang Shi, and Haoqi Fan. Emerging properties in unified multimodal pretraining. *arXiv preprint arXiv:2505.14683*, 2025.

[11] Ying Fan, Olivia Watkins, Yuqing Du, Hao Liu, Moonkyung Ryu, Craig Boutilier, Pieter Abbeel, Mohammad Ghavamzadeh, Kangwook Lee, and Kimin Lee. DPOK: Reinforcement learning for fine-tuning text-to-image diffusion models. *Advances in Neural Information Processing Systems*, 36:79858–79885, 2023.

[12] Dhruba Ghosh, Hannaneh Hajishirzi, and Ludwig Schmidt. GenEval: An object-focused framework for evaluating text-to-image alignment. *Advances in Neural Information Processing Systems*, 36:52132–52152, 2023.

[13] Google DeepMind. Gemini 3.1 Pro model card. `https://deepmind.google/models/model-cards/gemini-3-1-pro/`, February 2026.

[14] Helia Hashemi, Jason Eisner, Corby Rosset, Benjamin van Durme, and Chris Kedzie. LLM-Rubric: A multidimensional, calibrated approach to automated evaluation of natural language texts. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13806–13834, 2024.

[15] Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. CLIPScore: A reference-free evaluation metric for image captioning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7514–7528, 2021.

[16] Xiwei Hu, Rui Wang, Yixiao Fang, Bin Fu, Pei Cheng, and Gang Yu. ELLA: Equip diffusion models with LLM for enhanced semantic alignment. *arXiv preprint arXiv:2403.05135*, 2024.

[17] Yushi Hu, Reyhane Askari-Hemmat, Melissa Hall, Emily Dinan, Luke Zettlemoyer, and Marjan Ghazvininejad. Multimodal RewardBench 2: Evaluating omni reward models for interleaved text and image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 36904–36915, 2026.

[18] Mengzhao Jia, Zhihan Zhang, Ignacio Cases, Zheyuan Liu, Meng Jiang, and Peng Qi. AutoRubric: Rubric-based generative rewards for faithful multimodal reasoning. In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 25707–25724, 2026.

[19] Seungone Kim, Jamin Shin, Yejin Cho, Joel Jang, Shayne Longpre, Hwaran Lee, Sangdoo Yun, Seongjin Shin, Sungdong Kim, James Thorne, and Minjoon Seo. Prometheus: Inducing fine-grained evaluation capability in language models. In *The Twelfth International Conference on Learning Representations*, 2024.

[20] Yuval Kirstain, Adam Polyak, Uriel Singer, Shahbuland Matiana, Joe Penna, and Omer Levy. Pick-a-Pic: An open dataset of user preferences for text-to-image generation. *Advances in Neural Information Processing Systems*, 36:36652–36663, 2023.

[21] Tony Lee, Michihiro Yasunaga, Chenlin Meng, Yifan Mai, Joon Sung Park, Agrim Gupta, Yunzhi Zhang, Deepak Narayanan, Hannah Teufel, Marco Bellagente, et al. Holistic evaluation of text-to-image models. *Advances in Neural Information Processing Systems*, 36:69981–70011, 2023.

[22] Fan Li, Chonghuinan Wang, Lina Lei, Yuping Qiu, Jiaqi Xu, Jiaxiu Jiang, Xinran Qin, Zhikai Chen, Fenglong Song, Zhixin Wang, Renjing Pei, and Wangmeng Zuo. HP-Edit: A human-preference post-training framework for image editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 43113–43123, 2026.

[23] Zongjian Li, Zheyuan Liu, Qihui Zhang, Bin Lin, Feize Wu, Shenghai Yuan, Zhiyuan Yan, Yang Ye, Wangbo Yu, Yuwei Niu, Shaodong Wang, Xinhua Cheng, and Li Yuan. Uniworld-V2: Reinforce image editing with diffusion negative-aware finetuning and MLLM implicit feedback. *arXiv preprint arXiv:2510.16888v3*, 2025.

[24] Shiyu Liu, Yucheng Han, Peng Xing, Fukun Yin, Rui Wang, Wei Cheng, Jiaqi Liao, Yingming Wang, Honghao Fu, Chunrui Han, et al. Step1X-Edit: A practical framework for general image editing. *arXiv preprint arXiv:2504.17761*, 2025.

[25] Yixin Liu, Yue Yu, DiJia Su, Sid Wang, Xuewei Wang, Song Jiang, Bo Liu, Arman Cohan, Yuandong Tian, and Zhengxing Chen. Examining reasoning LLMs-as-judges in non-verifiable LLM post-training. In *Proceedings of the 2026 Conference on Empirical Methods in Natural Language Processing*, pages 56671–56704, 2026.

[26] Xin Luo, Jiahao Wang, Chenyuan Wu, Shitao Xiao, Xiyan Jiang, Defu Lian, Jiajun Zhang, Dong Liu, and Zheng Liu. EditScore: Unlocking online RL for image editing via high-fidelity reward modeling. In *The Fourteenth International Conference on Learning Representations*, 2026.

[27] Yiyang Ma, Xingchao Liu, Xiaokang Chen, Wen Liu, Chengyue Wu, Zhiyu Wu, Zizheng Pan, Zhenda Xie, Haowei Zhang, Xingkai Yu, Liang Zhao, Yisong Wang, Jiaying Liu, and Chong Ruan. JanusFlow: Harmonizing autoregression and rectified flow for unified multimodal understanding and generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7739–7751, 2025.

[28] Yuhang Ma, Xiaoshi Wu, Keqiang Sun, and Hongsheng Li. HPSv3: Towards wide-spectrum human preference score. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15086–15095, 2025.

[29] Tianjun Pan, Xuan Lin, Wenyan Yang, Qianyu He, Shisong Chen, Licai Qi, Wanqing Xu, Hongwei Feng, Bo Xu, and Yanghua Xiao. RubricEval: A rubric-level meta-evaluation benchmark for LLM judges in instruction following. *arXiv preprint arXiv:2603.25133*, 2026.

[30] Aditya Pathak, Rachit Gandhi, Vaibhav Uttam, Arnav Ramamoorthy, Pratyush Ghosh, Aaryan Raj Jindal, Shreyash Verma, Aditya Mittal, Aashna Ased, Chirag Khatri, Yashwanth Nakka, Devansh, Jagat Sesh Challa, and Dhruv Kumar. Rubric is all you need: Improving LLM-based code evaluation with question-specific rubrics. In *Proceedings of the 2025 ACM Conference on International Computing Education Research V. 1*, pages 181–195, 2025.

[31] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. SDXL: Improving latent diffusion models for high-resolution image synthesis. In *The Twelfth International Conference on Learning Representations*, 2024.

[32] Shelly Sheynin, Adam Polyak, Uriel Singer, Yuval Kirstain, Amit Zohar, Oron Ashual, Devi Parikh, and Yaniv Taigman. Emu Edit: Precise image editing via recognition and generation tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8871–8879, 2024.

[33] Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, et al. OpenAI GPT-5 system card. *arXiv preprint arXiv:2601.03267*, 2025.

[34] Bram Wallace, Meihua Dang, Rafael Rafailov, Linqi Zhou, Aaron Lou, Senthil Purushwalkam, Stefano Ermon, Caiming Xiong, Shafiq Joty, and Nikhil Naik. Diffusion model alignment using direct preference optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8228–8238, 2024.

[35] Peiyi Wang, Lei Li, Liang Chen, Zefan Cai, Dawei Zhu, Binghuai Lin, Yunbo Cao, Lingpeng Kong, Qi Liu, Tianyu Liu, and Zhifang Sui. Large language models are not fair evaluators. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9440–9450, 2024.

[36] Xinlong Wang, Xiaosong Zhang, Zhengxiong Luo, Quan Sun, Yufeng Cui, Jinsheng Wang, Fan Zhang, Yueze Wang, Zhen Li, Qiying Yu, Yingli Zhao, Yulong Ao, Xuebin Min, Tao Li, Boya Wu, Bo Zhao, Bowen Zhang, Liangdong Wang, Guang Liu, Zheqi He, Xi Yang, Jingjing Liu, Yonghua Lin, Tiejun Huang, and Zhongyuan Wang. Emu3: Next-token prediction is all you need, 2024.

[37] Yibin Wang, Zhimin Li, Yuhang Zang, Jiazi Bu, Yujie Zhou, Yi Xin, Junjun He, Chunyu Wang, Qinglin Lu, Cheng Jin, and Jiaqi Wang. UniGenBench++: A unified semantic evaluation benchmark for text-to-image generation. *arXiv preprint arXiv:2510.18701v2*, 2026.

[38] Yibin Wang, Zhimin Li, Yuhang Zang, Chunyu Wang, Qinglin Lu, Cheng Jin, and Jiaqi Wang. Unified multimodal chain-of-thought reward model through reinforcement fine-tuning. *Advances in Neural Information Processing Systems*, 38:176618–176645, 2025.

[39] Yibin Wang, Yuhang Zang, Hao Li, Cheng Jin, and Jiaqi Wang. Unified reward model for multimodal understanding and generation. *arXiv preprint arXiv:2503.05236*, 2025.

[40] Xinyu Wei, Jinrui Zhang, Zeqing Wang, Hongyang Wei, Zhen Guo, and Lei Zhang. TIIF-Bench: How does your T2I model follow your instructions? *arXiv preprint arXiv:2506.02161v2*, 2025.

[41] Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng-ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, et al. Qwen-Image technical report. *arXiv preprint arXiv:2508.02324*, 2025. Qwen-Image-Edit-2509 model card: `https://huggingface.co/Qwen/Qwen-Image-Edit-2509`.

[42] Chenyuan Wu, Jiahao Wang, Pengfei Zheng, Ruiran Yan, Shitao Xiao, Xin Luo, Yueze Wang, Wanli Li, Xiyan Jiang, Yexin Liu, Junjie Zhou, Ziyi Xia, Ze Liu, Chaofan Li, Haoge Deng, Kun Luo, Bo Zhang, Jiajun Zhang, Dong Liu, Defu Lian, Xinlong Wang, Zhongyuan Wang, Tiejun Huang, and Zheng Liu. OmniGen2: Towards instruction-aligned multimodal generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21964–21975, 2026.

[43] Keming Wu, Sicong Jiang, Max Ku, Ping Nie, Minghao Liu, and Wenhu Chen. EditReward: A human-aligned reward model for instruction-guided image editing. In *The Fourteenth International Conference on Learning Representations*, 2026.

[44] Xiaoshi Wu, Yiming Hao, Keqiang Sun, Yixiong Chen, Feng Zhu, Rui Zhao, and Hongsheng Li. Human preference score v2: A solid benchmark for evaluating human preferences of text-to-image synthesis. *arXiv preprint arXiv:2306.09341*, 2023.

[45] Jinheng Xie, Zhenheng Yang, and Mike Zheng Shou. Show-o2: Improved native unified multimodal models. *Advances in Neural Information Processing Systems*, 38:53002–53030, 2025.

[46] Lipeng Xie, Sen Huang, Zhuo Zhang, Anni Zou, Yunpeng Zhai, Dingchao Ren, Kezun Zhang, Haoyuan Hu, Boyin Liu, Haoran Chen, Zhaoyang Liu, and Bolin Ding. Auto-Rubric: Learning from implicit weights to explicit rubrics for reward modeling. *arXiv preprint arXiv:2510.17314*, 2025.

[47] Jiazheng Xu, Xiao Liu, Yuchen Wu, Yuxuan Tong, Qinkai Li, Ming Ding, Jie Tang, and Yuxiao Dong. ImageReward: Learning and evaluating human preferences for text-to-image generation. *Advances in Neural Information Processing Systems*, 36:15903–15935, 2023.

[48] Seonghyeon Ye, Doyoung Kim, Sungdong Kim, Hyeonbin Hwang, Seungone Kim, Yongrae Jo, James Thorne, Juho Kim, and Minjoon Seo. FLASK: Fine-grained language model evaluation based on alignment skill sets. In *The Twelfth International Conference on Learning Representations*, 2024.

[49] Yang Ye, Xianyi He, Zongjian Li, Bin Lin, Shenghai Yuan, Zhiyuan Yan, Bohan Hou, and Li Yuan. ImgEdit: A unified image editing dataset and benchmark. *Advances in Neural Information Processing Systems*, 38:146002–146032, 2025.

[50] Qifan Yu, Wei Chow, Zhongqi Yue, Kaihang Pan, Yang Wu, Xiaoyang Wan, Juncheng Li, Siliang Tang, Hanwang Zhang, and Yueting Zhuang. AnyEdit: Mastering unified high-quality image editing for any idea. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26125–26135, 2025.

[51] Kai Zhang, Lingbo Mo, Wenhu Chen, Huan Sun, and Yu Su. MagicBrush: A manually annotated dataset for instruction-guided image editing. *Advances in Neural Information Processing Systems*, 36:31428–31449, 2023.

[52] Xiangyu Zhao, Peiyuan Zhang, Junming Lin, Tianhao Liang, Yuchen Duan, Shengyuan Ding, Changyao Tian, Yuhang Zang, Junchi Yan, and Xue Yang. Trust your critic: Robust reward modeling and reinforcement learning for faithful image editing and generation. *arXiv preprint arXiv:2603.12247*, 2026.

[53] Kaiwen Zheng, Huayu Chen, Haotian Ye, Haoxiang Wang, Qinsheng Zhang, Kai Jiang, Hang Su, Stefano Ermon, Jun Zhu, and Ming-Yu Liu. DiffusionNFT: Online diffusion reinforcement with forward process. In *The Fourteenth International Conference on Learning Representations*, 2026.

Appendix

A Experimental setup details

This section provides a comprehensive account of the datasets, evaluation protocols, model configurations, training hyperparameters, and computational resources employed throughout the paper. All experiments were conducted on a cluster of 8 NVIDIA H100 (80GB SXM5) GPUs.

A.1 Datasets

We evaluate on two families of benchmarks: those designed for assessing preference evaluation fidelity, and those measuring generative quality in text-to-image synthesis and instruction-guided image editing.

Preference evaluation benchmarks.

- **HPDv3**: A large-scale human preference dataset for text-to-image generation comprising 1.17 million pairwise comparisons collected from diverse user prompts. Each pair presents two images generated from the same prompt, with one image annotated as preferred. We use the official test split and report pairwise preference accuracy.
- **MM-RewardBench2**: A diagnostic benchmark with 4,000 expert-annotated pairwise instances spanning four tasks: text-to-image alignment (T2I), image editing (Edit), visual question answering, and compositional understanding. We report accuracy on the T2I and Edit subtasks separately.
- **EditReward-Bench**: A fine-grained benchmark assessing instruction adherence in image editing, encompassing 13 subtasks with expert human annotations. Each subtask targets a distinct editing operation (e.g., object addition, texture transfer, style modification).

Generative benchmarks.

- **GenEval**: Assesses compositional object accuracy in T2I synthesis by verifying whether generated images contain correct objects and attributes as specified in the prompt. Accuracy is computed via object detection against structured prompt decompositions.
- **DPG-Bench**: Measures alignment with dense, paragraph-length prompts through structured question answering. We report the overall alignment score averaged across all test prompts.
- **TIIF**: Evaluates instruction fidelity across three difficulty tiers (simple, complex, compositional), providing a graded measure of instruction-following capacity. We report the macro-average across tiers.
- **UniGenBench++**: Probes semantic consistency with both Short and Long prompt variants, measuring coherence with brief versus detailed textual descriptions.
- **GEdit-Bench**: A real-world image editing benchmark comprising naturalistic user instructions. Outputs are evaluated by GPT-5 on a 1–10 scale covering instruction adherence, image quality, and preservation of non-targeted regions.
- **ImgEdit**: Evaluates single-turn and multi-turn instruction-driven editing quality using automated metrics and human assessments. We report the composite score averaged across categories and turn depths.

RL training datasets

- **ShareGPT-4o-Image**: A large-scale multimodal corpus for text-to-image generation and editing, containing around 92K high-quality GPT-4o-synthesized samples, including both text-to-image and text-guided image editing pairs, which we use to construct training and evaluation prompts.

A.2 Evaluation protocols

Preference accuracy. For all pairwise preference evaluators, we report *preference accuracy*: the proportion of test pairs for which the model assigns a higher reward (or preference) to the human-preferred image. To probe positional robustness, each test pair is evaluated in both its original (forward) and permuted (reverse) order. The gap between forward and reverse accuracy quantifies the degree of position bias.

Generative evaluation. For text-to-image generation, FLUX.1-dev uses sampling with 30 sampling steps and a guidance scale of 3.5. For image editing, Qwen-Image-Edit-2509 performs inference with 50 sampling steps and a classifier-free guidance (CFG) scale of 4.0. All benchmark evaluations are conducted using the official evaluation scripts without modification.

A.3 Model configurations

ARR instantiation. Unless otherwise specified, ARR employs a frozen VLM to synthesize five candidate rubrics per labeled training pair. The generation meta-prompt instructs the VLM to decompose the given text prompt into independent evaluative dimensions (e.g., object presence, attribute accuracy, spatial layout, aesthetic quality, instruction adherence), formulating each dimension as a verifiable binary criterion. Rubric synthesis, verification, and refinement are all conducted offline without gradient updates to the judge VLM.

For the ARR w/ guide variant, the meta-prompt is augmented with a fixed set of human-curated preference exemplars drawn from a held-out subset of the training split of each benchmark. These exemplars consist of (prompt, preferred image, dispreferred image, preference rationale) tuples and are embedded verbatim as in-context demonstrations. No fine-tuning of the VLM is performed; the exemplars serve solely as semantic anchors.

Rubric verification. Each candidate rubric r_i generated for a preference pair (x_i, y_i^+, y_i^-) is passed to a separate frozen verifier call, which checks whether applying r_i as a scoring criterion yields the correct preference decision on the generating pair. If verification fails, we invoke a refinement pass (up to $T_{\max} = 5$ iterations) that presents the verifier's critique alongside the original rubric and requests a revised formulation. Rubrics that remain unverified after $T_{\max}$ attempts are discarded. In our experiments, approximately 87% of initial rubrics pass verification without refinement, and fewer than 4% are ultimately discarded.

Hierarchical structuring. The verified rubric set $\mathcal{D}_R$ is organized into a hierarchical prompt structure by a final synthesis call. This call groups criteria by semantic level (coarse: overall alignment; mid: compositional attributes; fine: local details) and orders them by estimated diagnostic value. The resulting structured rubric $R_{\text{structured}}$ is formatted as a numbered list of axis definitions, each accompanied by a brief operationalization clause. This structure is passed verbatim to the judge VLM as the evaluation conditioning context.

A.4 Generative training: RPO hyperparameters

RPO is a reinforcement learning algorithm adapted for denoising diffusion policies in both text-to-image generation (T2I) and image editing. Key hyperparameters are reported in Table 3.

Training prompts are sampled uniformly from ShareGPT4o-Image, with no data augmentation applied. At each online iteration, two candidate outputs are generated from the current policy π_θ, evaluated by the frozen ARR judge conditioned on the structured rubric, and the resulting binary advantage $A \in \{+\lambda, -\gamma\}$ is uniformly distributed across all generation timesteps.

B Auto-Rubric as Reward (ARR) details

This section elaborates on the technical instantiation of Auto-Rubric as Reward (ARR), complementing the concise description provided in Section 3.2 of the main text. We provide a granular account of the rubric generation pipeline, the verification and refinement protocol, the hierarchical structuring mechanism, and a comparative characterization of ARR within the broader landscape of reward modeling approaches.

Table 3: RPO training hyperparameters for T2I (FLUX.1.dev) and image editing (Qwen-Image-Edit-2509).

Hyperparameter	**T2I (FLUX.1.dev)**	**Editing (Qwen-Image-Edit)**
Base learning rate	5×10^{-5}	1×10^{-5}
Batch size	32	16
Candidates per prompt	2	2
Clip ϵ	0.2	0.2
KL coefficient β	0.01	0.02
Positive reward λ	1.0	1.0
Negative reward γ	0.1	0.1
Denoising steps	8	10
Optimizer	AdamW	AdamW
Gradient clipping	1.0	1.0
Trained Parameters	LoRA (rank=16)	LoRA (rank=32)

B.1 Rubric generation pipeline

ARR synthesizes prompt-conditioned rubrics through a three-stage process: *generation*, *verification*, and *structuring*. Each stage is implemented as a frozen (multimodal) large language model call, ensuring that the judge VLM remains unmodified throughout.

B.1.1 Per-instance rubric generation

Given a preference pair (x, y^+, y^-) drawn from a pairwise dataset $\mathcal{D}$, we prompt the generator model $\mathcal{M}_{\text{gen}}$ to produce a natural language explanation of why y^+ is preferred over y^-. The meta-prompt explicitly instructs the model to:

- Decompose the preference into independent, verifiable quality axes (e.g., semantic fidelity, attribute accuracy, spatial coherence).
- Formulate each axis as a binary criterion that can be evaluated without reference to the paired candidate.
- Avoid holistic or comparative language that presupposes knowledge of both outputs.

The resulting rubric r_i is a structured, axis-wise decomposition of the preference rationale.

B.1.2 Verification and refinement

Each candidate rubric r_i is validated by a separate verifier call $\mathcal{M}_{\text{verify}}$. The verifier receives the original preference pair (x, y^+, y^-) and the generated rubric r_i, and is tasked with determining whether applying r_i as an evaluation protocol correctly identifies y^+ as the preferred output. The verification outcome is binary:

$$v_i = \begin{cases} \text{true} & \text{if } \mathcal{M}_{\text{verify}}(x, y^+, y^-, r_i) \text{ confirms the original preference,} \\ \text{false} & \text{otherwise.} \end{cases}$$

If verification fails, we invoke a refinement pass that supplies the verifier's critique alongside r_i to a refinement model $\mathcal{M}_{\text{refine}}$, which produces a revised rubric $r_i^{(t+1)}$. Refinement iterates up to $T_{\max} = 5$ times; rubrics that remain unverified after this budget are discarded. Empirically, 87% of initial rubrics pass verification without refinement, and fewer than 4% are ultimately discarded, attesting to the stability of the generation process.

B.1.3 Hierarchical structuring

The verified rubric collection $\mathcal{D}_R = \{r_i \mid v_i = \text{true}\}$ is subsequently aggregated into a single, hierarchically structured prompt $R_{\text{structured}}$. The structuring model $\mathcal{M}_{\text{struct}}$ organizes the rubrics into different evaluation dimensions, including:

Table 4: **Paradigm shift from implicit to explicit reward parameterization.** Comparison of multimodal reward modeling approaches along key operational axes. Pointwise and pairwise reward models require extensive preference data and yield opaque scalar signals. Mix refers to models that support both pairwise and pointwise outputs. ARR uniquely combines zero-shot rubric generation with binary scoring, eliminating training overhead entirely.

Method	Mode	Reward Form	Hacking Risk	Interpretability	Data Requirement
PickScore	Pointwise	Scalar	High	Low	Large
HPSv3	Pointwise	Scalar	High	Low	Large
ImageReward	Pointwise	Scalar	Medium	Low	Large
UnifiedReward	Mix	Scalar	Medium	Low	Large
VLM-as-Judge	Pairwise (Mix)	Binary	Medium	Medium	Zero-shot
ARR (Ours)	Pairwise (Mix)	Binary	Low	High	Zero-shot

- **Overall alignment**: Measures the global consistency between the generated output and the prompt intent.
- **Compositional structure**: Evaluates the presence and relationships of key elements, such as object presence and spatial relations.
- **Fine-grained fidelity**: Focuses on local details and editing-specific accuracy.
- **Other dimensions**: ...

The final structured rubric is formatted as a numbered list of evaluation dimensions, where each dimension groups a set of corresponding rubrics. For each dimension, a brief operationalization clause is provided to clarify its focus. This structured rubric is directly used as the conditioning context for the judge VLM during both evaluation and RPO training.

B.2 Comparative characterization of reward modeling paradigms

Table 4 situates ARR within the landscape of contemporary multimodal reward modeling. We contrast ARR against representative pointwise reward models, pairwise reward models, and direct VLM judges. The comparison spans five dimensions: evaluation mode, reward representation, susceptibility to reward hacking, interpretability of the reward signal, and data requirements for training or deployment.

Key distinctions. ARR differs from prior approaches in four critical respects:

1. **Zero-shot rubric generation**: ARR synthesizes rubrics on-the-fly from frozen VLMs, enabling immediate deployment in new domains without additional data collection or task-specific supervision.
2. **Holistic, rubric-conditioned decision interface**: Rather than aggregating independently scored criteria post hoc, ARR formulates evaluation as a single rubric-conditioned judgment, where all dimensions are jointly considered in a pairwise comparison. This preserves inter-criterion dependencies and avoids inconsistencies introduced by independent scoring and aggregation.
3. **Training-free reward interface**: ARR operates without any parameter updates to the judge model, eliminating the computational and data overhead associated with training pointwise or pairwise reward models, while retaining strong generalization through the underlying VLM.
4. **Data-efficient rubric induction**: Across all experiments, high-quality rubrics are constructed from as few as 100 preference pairs drawn from ShareGPT-4o-Image. This demonstrates that ARR can recover structured, task-relevant evaluation criteria with minimal supervision, achieving competitive performance with substantially lower data requirements than existing methods.

These properties collectively establish ARR as a lightweight, interpretable, and bias-resilient alternative to both implicit reward models and manually curated rubric systems. Importantly, since ARR

builds on a VLM-as-a-judge paradigm, the rubric-conditioned interface is inherently flexible and can be extended beyond pairwise comparison to pointwise scoring or listwise ranking settings. In this work, we focus on the pairwise formulation to isolate and evaluate the robustness of rubric-based decision making under minimal reward hacking risk, providing a controlled setting for studying structured, generative reward modeling in multimodal alignment.

C Ablations on position bias in ARR

C.1 Setup

Position bias refers to the systematic tendency of a pairwise preference evaluator to favor whichever candidate appears in a fixed ordinal position (e.g., always preferring Image A when presented first), irrespective of actual quality. This constitutes a critical failure mode: an evaluator that achieves high accuracy in the standard presentation order but collapses under permutation produces a spurious reward signal entangled with input ordering rather than genuine quality.

To isolate this effect, we evaluate each pairwise evaluator on the HPDv3 test set under two conditions: (i) *forward order*, where images appear in the original benchmark order; and (ii) *reverse order*, where the two images are swapped. We report forward accuracy (%), reverse accuracy (%), and their arithmetic mean (Avg). An ideal, unbiased evaluator would achieve identical accuracy under both conditions. The quantity $\Delta = \text{Acc}_{\text{fwd}} - \text{Acc}_{\text{rev}}$ serves as our primary measure of positional instability; larger Δ indicates stronger position bias.

C.2 Results

Table 5 reports the position bias ablation across three base VLMs and their ARR-augmented variants. All experiments are conducted on the HPDv3 test set.

Table 5: **Position bias ablation on HPDv3.** Forward and reverse accuracy (%) are measured by swapping the order of the two images in each preference pair. $\Delta = \text{Fwd} - \text{Rev}$ quantifies positional instability. ARR variants reduce Δ consistently; ARR w/ guide provides the strongest stabilization. Rows are grouped by base model.

Method	Forward	Reverse	Avg	Δ
Qwen3-VL-8B	84.5	49.9	67.2	34.6
+ ARR	86.0	54.4	70.2	31.6
+ ARR w/ guide	**90.1**	**79.8**	**85.0**	10.3
GPT-5	88.7	56.1	72.4	32.6
+ ARR	90.2	62.0	76.1	28.2
+ ARR w/ guide	**93.4**	**84.1**	**88.8**	9.3
Gemini 3.1 Pro	91.7	61.5	76.6	30.2
+ ARR	92.2	64.4	78.3	27.8
+ ARR w/ guide	**95.2**	**86.3**	**90.8**	8.9

C.3 Analysis

Table 5 reveals four consistent patterns:

Base VLMs exhibit severe and scale-invariant position bias. Across all three base models, the gap between forward and reverse accuracy is extreme: $\Delta = 34.6$ for Qwen3-VL-8B, 32.6 for GPT-5, and 30.2 for Gemini 3.1 Pro. Crucially, this gap does not diminish with model capability: the most capable model (Gemini 3.1 Pro) yields a marginally smaller but still operationally severe Δ of 30.2. This confirms that positional instability is a structural deficiency tied to the implicit parameterization of preference knowledge, not a capacity limitation that resolves with scale.

Zero-shot ARR yields consistent but moderate debiasing. Conditioning the VLM on auto-generated rubrics reduces Δ by 3.0–4.8 points across all three models (e.g., $34.6 \rightarrow 31.6$ for Qwen3-VL-8B). The mechanism is interpretable: by requiring the model to commit to explicit evaluation criteria before inspecting the candidates, ARR partially anchors the judgment in criterion-level evidence rather than holistic gestalt impressions susceptible to ordering heuristics. However, a substantial gap persists, indicating that self-generated rubrics alone do not fully overcome the structural mismatch between latent preference encoding and stable pairwise judgment.

Preference-conditioned ARR provides qualitatively stronger stabilization. `ARR w/ guide` reduces Δ dramatically, to 10.3, 9.3, and 8.9 for the three base models respectively, corresponding to reductions of 24.3, 23.3, and 21.3 points relative to the unaugmented baseline. The effect on reverse accuracy is particularly striking: Qwen3-VL-8B improves from 49.9% (near-random on reversed pairs) to 79.8%, indicating that human preference exemplars substantially enhance the model's capacity to identify quality differences in an order-agnostic manner. This suggests that the key failure mode in unaugmented VLM judges is not an inability to perceive relevant features, but rather an inability to stably weight them independently of presentation order, a failure that explicit, human-grounded rubrics can partially correct.

Residual bias remains non-trivial. Even under `ARR w/ guide` with Gemini 3.1 Pro ($\Delta = 8.9$), meaningful positional instability persists. A perfectly unbiased evaluator would achieve $\Delta = 0$. This residual gap underscores that current VLMs do not yet fully ground preference evaluation in stable criteria, and that stronger human preference guidance amplifies the effect of ARR rather than eliminating the need for it.

C.4 Cross-model rubric transfer

Table 6: Cross-model rubric transfer on HPDv3. The judge is fixed to Gemini 3.1 Pro; only the rubric generator varies. The direct baseline uses no rubric. Accuracy (%) denotes agreement with human preference labels.

Rubric Generator	Judge	Accuracy (%)
None (direct)	Gemini 3.1 Pro	75.9
Qwen3-VL-8B	Gemini 3.1 Pro	77.5
GPT-5	Gemini 3.1 Pro	78.3
Gemini 3.1 Pro	Gemini 3.1 Pro	79.2

Cross-model rubric transfer. To further verify that ARR does not rely on same-family co-adaptation, we fix Gemini 3.1 Pro as the judge and generate rubrics using Qwen3-VL-8B, GPT-5, and Gemini 3.1 Pro. Table 6 reports HPDv3 accuracy from a transfer experiment conducted separately from the evaluations in Tables 1 and 5. Even rubrics from the weakest generator, Qwen3-VL-8B, improve accuracy from 75.9% (direct) to 77.5%, closing more than half of the gap to same-family rubrics (79.2%). This demonstrates that the rubric structure itself, rather than shared model biases, is the primary contributor to evaluation robustness.

D Ablations on rubric cardinality

D.1 Setup

The number of candidate rubrics generated offline per labeled training pair represents a key design choice in ARR. Too few rubrics may underspecify the relevant evaluation space, failing to capture important axes of quality; too many may introduce redundant, conflicting, or noisy criteria that degrade the signal-to-noise ratio of the resulting reward. To systematically study this trade-off, we vary the number of rubrics generated per item ($K \in \{1, 5, 10, 20\}$) while applying the same hierarchical structuring step to all settings, and measure preference accuracy on the HPDv3 test set. All experiments employ Qwen3-VL-8B-Instruct as the base judge and utilize zero-shot rubric generation (i.e., without human preference exemplars), thereby isolating the effect of cardinality from that of guidance quality.

D.2 Results

Table 7 reports preference accuracy as a function of rubric cardinality. Accuracy is reported as the average of forward and reverse evaluation conditions to ensure that gains are not confounded by positional bias.

Table 7: **Rubric cardinality vs. preference accuracy (HPDv3, Qwen3-VL-8B-Instruct, ARR zero-shot).** Accuracy (%) reported as average of forward and reverse conditions. $K = 5$ is used as the default in all main experiments.

Num. Rubrics per Item (K)	**HPDv3 Accuracy (%)**
1	69.8
5	70.2
10	72.1
20	74.4

D.3 Analysis

Accuracy improves monotonically with rubric cardinality. Increasing K from 1 to 20 yields a consistent improvement from 69.8% to 74.4%, a net gain of 4.6 percentage points. This monotonic trend indicates that additional rubric dimensions provide genuinely complementary information rather than merely redundant coverage: each additional axis captures aspects of quality that are not fully addressed by fewer criteria, leading to more discriminative and robust evaluations.

The gains from $K = 1$ to $K = 5$ are modest but non-negligible. The increment from a single rubric to five rubrics yields only 0.4 percentage points, suggesting that a well-formed single rubric already captures the primary quality axis relevant to a given preference pair. However, the subsequent gains from $K = 5$ to $K = 10$ ($+1.9$ points) and from $K = 10$ to $K = 20$ ($+2.3$ points) demonstrate that finer-grained decomposition becomes increasingly consequential for difficult pairs where the quality differential is subtle or multidimensional.

Practical trade-offs and the choice of $K = 5$. A larger K increases offline generation and verification cost. Each ordered pair presentation uses one judge call with the cached task rubric. We find that $K = 5$ provides a favorable accuracy–efficiency trade-off, achieving 70.2% accuracy with modest offline computational overhead. For deployment contexts where the offline rubric construction budget is constrained, $K = 5$ constitutes a well-calibrated default. For high-stakes evaluation scenarios, $K = 20$ delivers the strongest performance in this ablation.

Noise considerations at high cardinality. We note that despite the accuracy gains, larger K also elevates the probability of including noisy or redundant criteria: the marginal rubric at $K = 20$ is necessarily less discriminative than the most salient rubric at $K = 1$. In RPO training, noisy rubric axes contribute low-magnitude gradient signals whose impact is diluted through averaging across axes during reward aggregation, without harming overall convergence.

E Rubric Policy Optimization details

E.1 Algorithm overview

RPO is an online policy gradient algorithm that leverages ARR-generated rubrics as binary reward signals to align a generative policy π_θ with multidimensional human preferences. As described in Section 3.3 of the main text, RPO operates in a fully online fashion: at each iteration, it (i) samples prompts from a training distribution $\mathcal{D}$, (ii) generates two candidate outputs from the current policy, (iii) evaluates the candidates using the frozen ARR judge conditioned on the cached task rubric, and (iv) updates π_θ via a PPO-style policy gradient with KL regularization. The full training procedure is formalized in Algorithm 1.

Algorithm 1 Rubric Policy Optimization (RPO)

Require: Pretrained policy π_{θ_0}, reference policy π_{ref}, frozen ARR judge $\mathcal{M}_\theta$, training prompt distribution $\mathcal{D}$, number of iterations N, batch size B, positive reward magnitude λ, negative reward magnitude γ, Clip threshold ϵ, KL coefficient β

Ensure: Optimized policy π_{θ_N}

for iteration $k = 1, \ldots, N$ **do**
  Sample a batch of prompts $\{h_j\}_{j=1}^{B}$ from $\mathcal{D}$
  for each prompt h_j **do**
    Generate two candidate outputs: $y_j^1, y_j^2 \sim \pi_{\theta_{k-1}}(\cdot \mid h_j)$
    Set $R_j \leftarrow R_{\text{structured}}$, where $R_{\text{structured}}$ is the fixed cached task rubric {via ARR (Section 3.2)}
    Obtain binary preference: $p_j = \mathcal{M}_\theta(y_j^1 \succ y_j^2 \mid h_j, R_j)$
    Assign advantages: $A_j^{\text{win}} \leftarrow +\lambda$, $A_j^{\text{loss}} \leftarrow -\gamma$
    Distribute advantage uniformly across all generation timesteps $t = 0, \ldots, T-1$
  end for
  Compute PPO-clipped objective $\mathcal{L}_{\text{RPO}}(\theta_{k-1})$ according to Equation (12)
  Update policy: $\theta_k \leftarrow \theta_{k-1} + \eta \nabla_\theta \mathcal{L}_{\text{RPO}}(\theta_{k-1})$
end for
return π_{θ_N}

E.2 KL regularization and training stability

The KL penalty $\beta D_{\text{KL}}(\pi_\theta \| \pi_{\text{ref}})$ in the RPO objective (Equation 12 in the main text) serves two purposes. First, it prevents excessive policy drift away from the pretrained reference distribution π_{ref}, preserving the generative priors learned during pretraining. Second, it stabilizes training by bounding the entropy reduction induced by reward maximization, thereby mitigating mode collapse. We set $\beta = 0.01$ for T2I and $\beta = 0.02$ for image editing; the higher editing coefficient reflects the narrower action space and greater susceptibility to distributional collapse.

We observe that RPO training exhibits substantially lower variance in reward trajectories compared to reward-model-based RL baselines. We attribute this stability to two factors: (i) the frozen nature of the ARR judge eliminates reward model drift as a source of instability, and (ii) the rubric-conditioned binary signal provides a more consistent gradient direction than scalar reward models, which collapse multi-dimensional quality into a single value subject to distributional shift.

F Limitation

Frozen-model focus and fine-tuning potential. This work deliberately concentrates on frozen multimodal foundation models to isolate the effect of externalizing latent preference knowledge through auto-generated rubrics, rather than through parameter updates. By converting implicit, entangled preference representations into explicit, independently verifiable criteria, these rubrics serve as a structured interface that systematically suppresses evaluation biases, enhances interpretability, and directly translates into more reliable and hack-resistant reward signals for generative alignment. The demonstration that this training-free rubric conditioning alone can outperform dedicated pairwise reward models underscores the overlooked significance of the preference interface itself: the primary bottleneck in multimodal alignment is not a deficit of model capacity, but the absence of a stable, factorized criterion space for applying it. While further fine-tuning of the underlying VLMs would likely improve rubric fidelity and downstream generative quality, our results establish that the core value of the rubric paradigm lies in making preference evaluation structurally transparent, bias-resilient, and scalable across tasks without requiring any modification to the judge model.

Pairwise formulation as a robustness measure. We adopt pairwise comparison as the core evaluation protocol because its comparative nature offers stronger structural resistance to reward hacking than pointwise scoring or differentiable reward models. Conditioning these judgments on explicit, prompt specific rubrics amplifies this resilience by grounding evaluation in inspectable, independently verifiable criteria that leave little room for opaque manipulation. This rubric driven interface also endows ARR with considerable extensibility: because the criteria are expressed in

natural language, they can be dynamically expanded, refined, or adapted to new domains without any retraining of the underlying judge. The rubric thus functions as a transparent, stable scaffold that decouples evaluation logic from model parameters, preserving interpretability even as task requirements evolve. Although generative reward models and end to end VLM judges offer alternative paradigms, we prioritize the pairwise rubric interface as the most defensible, interpretable, and hack resistant configuration within the current alignment landscape, precisely because it externalizes the preference structure that other approaches leave implicit.

Human supervision and self-improvement. While the ARR framework readily accommodates human supervision to further refine rubric quality and specificity, the present work deliberately emphasizes what can be achieved with no additional annotation. Our experiments demonstrate that multimodal foundation models can substantially self-improve their comprehension and reasoning over human preferences purely through the auto-rubric process, using only self-generated criteria to guide evaluation and optimization. This finding is significant because it reveals that the rubric mechanism itself, even without curated exemplars, provides a sufficient and scalable structure for preference alignment, transforming latent knowledge into actionable, verifiable constraints. Nevertheless, we acknowledge that fully automated rubric generation may not yet reach the precision or domain-specific nuance that curated human guidance could provide, and we therefore treat the deeper integration of human-in-the-loop rubric curation as a natural and valuable extension. The present results establish a lower bound: even in the absence of human intervention, externalizing preference structure through auto-rubrics proves remarkably effective, while the upper bound accessible through human refinement remains an open and promising direction.

G Image generation and editing examples



Figure 5: **Examples of text-to-image generation.**



Figure 6: **Examples of image editing.**

H Full results

H.1 Image generation and editing

Table 8: **Generative performance across T2I and image editing benchmarks.** Blue-shaded rows denote ARR-RPO. Green arrows indicate absolute gains over the baseline.

Method	Text-to-Image					Image Editing	
	GenEval	DPG-Bench	TIIF	UniGenBench++		GEdit-Bench	ImgEdit
				Short	Long		
Specialist Model (T2I)							
SDXL[31]	0.55	74.65	54.96	40.22	41.48	—	—
Emu3[36]	0.54	80.60	–	45.42	50.59	—	—
JanusFlow[27]	0.63	79.68	–	47.10	54.80	—	—
FLUX.1-Dev	0.66	83.84	71.09	60.97	69.42	—	—
DALLE-3[3]	0.67	83.50	74.96	68.85	70.82	—	—
BLIP3o-4B[7]	0.81	79.36	–	59.57	61.01	—	—
Janus-Pro-7B[9]	0.80	84.19	66.50	61.36	71.11	—	—
Show-o2[45]	0.76	86.14	–	61.90	70.33	—	—
OmniGen2[42]	0.80	83.57	–	63.09	71.39	—	—
BAGEL[10]	0.82	85.07	71.50	59.91	71.26	—	—
ARR-RPO / T2I (Ours)							
w/ RPO-Qwen3vl-8B	0.72 ↑0.06	84.67 ↑0.83	73.81 ↑2.72	63.28 ↑2.31	71.05 ↑1.63	—	—
w/ RPO-Qwen3vl-8B-ARR	0.74 ↑0.08	85.03 ↑1.19	74.92 ↑3.83	64.17 ↑3.20	71.82 ↑2.40	—	—
w/ RPO-GPT-5	0.76 ↑0.10	84.97 ↑1.13	74.84 ↑3.75	64.22 ↑3.25	71.78 ↑2.36	—	—
w/ RPO-GPT-5-ARR	0.78 ↑0.12	85.41 ↑1.57	76.18 ↑5.09	65.36 ↑4.39	72.41 ↑2.99	—	—
w/ RPO-Gemini 3.1 Pro	0.77 ↑0.11	85.02 ↑1.18	75.69 ↑4.60	64.76 ↑3.79	72.13 ↑2.71	—	—
w/ RPO-Gemini 3.1 Pro-ARR	0.80 ↑0.14	85.76 ↑1.92	76.85 ↑5.76	65.89 ↑4.92	72.93 ↑3.51	—	—
Specialist Model (Editing)							
Instruct-Pix2Pix[6]	—	—	—	—	—	3.68	1.88
AnyEdit[50]	—	—	—	—	—	3.21	2.45
Step1X-Edit[24]	—	—	—	—	—	6.97	3.06
Qwen-Image-Edit-2509[41]	—	—	—	—	—	7.54	4.35
UniWorld-Qwen-Image-Edit[23]	—	—	—	—	—	7.76	4.48
ARR-RPO / Image Editing (Ours)							
w/ RPO-Qwen3vl-8B	—	—	—	—	—	7.63 ↑0.09	4.37 ↑0.02
w/ RPO-Qwen3vl-8B-ARR	—	—	—	—	—	7.66 ↑0.12	4.38 ↑0.03
w/ RPO-GPT-5	—	—	—	—	—	7.65 ↑0.11	4.38 ↑0.03
w/ RPO-GPT-5-ARR	—	—	—	—	—	7.72 ↑0.18	4.40 ↑0.05
w/ RPO-Gemini 3.1 Pro	—	—	—	—	—	7.79 ↑0.25	4.39 ↑0.04
w/ RPO-Gemini 3.1 Pro-ARR	—	—	—	—	—	7.85 ↑0.31	4.43 ↑0.08

Table 9: **Additional experimental results: Post-training performance of BAGEL using ARR-RPO across T2I benchmarks.** Best in **bold**. Blue-shaded rows denote ARR-RPO variants. Green arrows indicate absolute gains over the BAGEL baseline.

Method	GenEval	DPG-Bench	TIIF	UniGenBench++	
				Short	Long
BAGEL (Baseline)					
BAGEL[10]	0.82	85.07	71.50	59.91	71.26
ARR-RPO / T2I (Ours)					
w/ RPO-Qwen3vl-8B	0.85 ↑0.03	85.45 ↑0.38	73.12 ↑1.62	62.34 ↑2.43	71.89 ↑0.63
w/ RPO-Qwen3vl-8B-ARR	0.88 ↑0.06	85.82 ↑0.75	74.05 ↑2.55	63.05 ↑3.14	72.35 ↑1.09
w/ RPO-GPT-5	0.86 ↑0.04	85.91 ↑0.84	74.48 ↑2.98	63.72 ↑3.81	72.58 ↑1.32
w/ RPO-GPT-5-ARR	0.90 ↑0.08	86.28 ↑1.21	75.62 ↑4.12	64.81 ↑4.90	73.15 ↑1.89
w/ RPO-Gemini 3.1 Pro	0.87 ↑0.05	86.15 ↑1.08	75.41 ↑3.91	64.62 ↑4.71	72.97 ↑1.71
w/ RPO-Gemini 3.1 Pro-ARR	**0.92** ↑0.10	**86.74** ↑1.67	**76.85** ↑5.35	**65.92** ↑6.01	**73.82** ↑2.56

H.2 Human preference

Table 10: **Evaluator performance across four preference benchmarks.** Accuracy (%) denotes agreement with human preference labels. The best result in each column is **bold**. Blue-shaded rows indicate ARR. Green arrows indicate absolute gains over the baseline.

Method	HPDv3	MM-RewardBench2 (T2I)	MM-RewardBench2 (Edit)	EditReward-Bench
	Acc.	Acc.	Acc.	Acc.
Trained Reward Model				
CLIPScore[15]	48.6	51.0	—	—
PickScore[20]	65.6	58.6	—	—
ImageReward[47]	58.6	54.0	—	—
UnifiedReward[39]	66.0	59.8	—	—
UnifiedReward-Thinking[38]	68.1	66.0	—	—
HPSv2[44]	65.3	54.7	—	—
HPSv3[28]	76.9	60.2	—	—
EditReward[43]	—	—	67.2	56.45
VLM-as-Judge (Direct)				
Qwen3-VL-8B	67.2	57.6	59.2	54.01
GPT-5	72.4	70.5	73.8	57.53
Gemini 3.1 Pro	76.6	75.1	77.4	61.23
ARR (Ours)				
Qwen3vl-8B + ARR	70.2 (± 0.2) ↑3.0	62.7 (± 0.2) ↑5.1	65.5 (± 0.3) ↑6.3	57.22 (± 0.1) ↑3.21
GPT-5 + ARR	76.1 (± 0.2) ↑3.7	74.7 (± 0.4) ↑4.2	77.5 (± 0.3) ↑3.7	61.01 (± 0.1) ↑3.48
Gemini 3.1 Pro + ARR	**78.3** (± 0.1) ↑1.7	**78.9** (± 0.2) ↑3.8	**79.2** (± 0.2) ↑1.8	**63.27** (± 0.2) ↑2.04

I Prompts and rubrics

T2I rubrics (Gemini 3.1 Pro)

```
Rubric 1:
Theme: Architectural and Structural Fidelity
- Tip1: Assess clarity and precision of building facades, windows, railings, and
    structural elements for geometric accuracy and absence of
distortion.
- Tip2: Evaluate consistency of architectural details (e.g., columns, arches, textures)
    across the scene without illogical or impossible
geometry.
- Tip3: Check for realistic material rendering (wood grain, stone patina, metal sheen)
    and absence of overly smooth or artificial surfaces.
- Tip4: Ensure environmental context is coherent, with buildings, paths, and landscapes
    logically integrated and proportionally accurate.
Rubric 2:
Theme: Lighting, Shadow, and Reflection Realism
- Tip1: Evaluate how naturally light interacts with surfaces, including direction,
    intensity, and shadow gradients that match the scene's
source.
- Tip2: Assess reflections on water, glass, and metallic surfaces for physical
    plausibility and consistency with ambient lighting.
- Tip3: Check for believable depth and dimensionality through consistent, directional
    shadows and highlights across all objects and subjects.
- Tip4: Identify flat, unnatural, or inconsistent lighting that flattens the scene or
    creates artificial glow and highlights.
Rubric 3:
Theme: Texture and Material Fidelity
- Tip1: Judge the clarity and detail of fine textures (fabric weave, fur, wood grain,
    skin pores) without smearing or over-smoothing.
- Tip2: Assess realism of material properties (metallic sheen, fabric wear, plastic gloss
    ) and absence of uniform, plastic-like surfaces.
- Tip3: Look for natural imperfections (dust, scratches, frayed edges) that enhance
    authenticity and avoid overly pristine or artificial
finishes.
- Tip4: Evaluate surface variations and non-repetitive patterns to distinguish between
    realistic textures and AI-generated uniformity.
Rubric 4:
Theme: Anatomical and Object Plausibility
- Tip1: Assess anatomical accuracy of human and animal forms, including proportions,
    posture, and facial features for naturalism.
- Tip2: Evaluate mechanical and structural plausibility of objects (tools, vehicles,
    clothing) for logical construction and physical
consistency.
- Tip3: Check for distorted limbs, unnatural hand shapes, or illogical object boundaries
    that break immersion or suggest AI artifacts.
- Tip4: Ensure subjects interact with their environment in believable ways (e.g., shadows
    , occlusion, posture) without surreal or impossible
elements.
Rubric 5:
Theme: Visual Coherence and Composition
- Tip1: Assess overall scene balance, depth, and arrangement of elements for aesthetic
    harmony and intentional design.
- Tip2: Evaluate framing, perspective, and depth of field for natural spatial
    relationships and absence of awkward or cluttered compositions.
- Tip3: Judge the integration of foreground, midground, and background for visual flow
    and environmental immersion.
- Tip4: Look for compositional clarity, with focal points and visual hierarchy that guide
     the viewer's eye without distraction.
Rubric 6:
Theme: AI Artifact Detection and Elimination
- Tip1: Identify telltale signs of AI generation: unnatural edges, repeated patterns,
    inconsistent perspective, or surreal details.
- Tip2: Look for artifacts such as smearing, halos, pixelation, or warped geometry that
    disrupt visual authenticity.
```

T2I rubrics (Gemini 3.1 Pro) (continued)

```
- Tip3: Assess for illogical object placement, duplicated elements, or inconsistent
    lighting that break scene coherence.
- Tip4: Prioritize images with minimal or no AI artifacts, especially in critical areas
    like anatomy, textures, and object boundaries.
Rubric 7:
Theme: Environmental and Background Realism
- Tip1: Evaluate the realism of natural elements (trees, water, clouds, terrain) for
    accurate physics, textures, and atmospheric perspective.
- Tip2: Assess background depth and context for logical scale, perspective, and
    integration with foreground subjects.
- Tip3: Check for believable environmental details (e.g., foliage, ground texture, sky
    gradients) without AI-style repetition or over-
saturation.
- Tip4: Ensure background elements (buildings, landscapes, props) are contextually
    plausible and not visually disconnected from the scene.
Rubric 8:
Theme: Text, Typography, and Detail Legibility
- Tip1: Assess clarity and legibility of text, including font consistency, spacing, and
    absence of smudging or distortion.
- Tip2: Evaluate spelling accuracy, grammar, and semantic coherence of written elements (
    e.g., signs, captions, product labels).
- Tip3: Check for clean, well-defined typography with appropriate contrast and alignment
    against background elements.
- Tip4: Prioritize images with crisp, readable text and no garbled, pixelated, or AI-
    generated text artifacts.
Rubric 9:
Theme: Color, Gradient, and Aesthetic Harmony
- Tip1: Assess color palette richness, saturation, and harmony for contextually
    appropriate and visually pleasing transitions.
- Tip2: Evaluate natural lighting and color grading for believable gradients, shadows,
    and reflections without artificial neon or
oversaturation.
- Tip3: Judge overall aesthetic appeal, including balance, contrast, and intentional
    visual design that enhances mood or narrative.
- Tip4: Look for unified color schemes that support the scene's theme without jarring or
    mismatched tones.
Rubric 10:
Theme: Detail Fidelity and Micro-Texture Rendering
- Tip1: Assess sharpness and clarity of fine details (hair strands, stitching,
    buttonholes, tool engravings) without AI-style blurring.
- Tip2: Evaluate preservation of intricate textures (fur, fabric, metal, skin) that
    convey realism and depth without over-smoothing.
- Tip3: Check for consistent rendering of small elements (e.g., droplets, dust, scratches
    ) that add authenticity without artificial
repetition.
- Tip4: Prioritize images where micro-textures are naturally varied and physically
    plausible, not uniformly or algorithmically generated.
```

Figure 7: **Auto-generated T2I rubrics (Gemini 3.1 Pro).** Example prompt-conditioned rubrics automatically synthesized by ARR for text-to-image evaluation, spanning dimensions such as architectural fidelity, lighting consistency, texture realism, and AI artifact detection.

T2I evaluation system prompt

```
Based on the given evaluation rubrics and the caption, please perform a **pairwise
comparison** between the two input images (Image 1 and Image 2) to determine which one
    has
higher quality.
{task_description_section}
Evaluation Rubrics:
{rubrics}
**Caption: {sample_content}**
## Task Requirements
- You are evaluating two images: Image 1 and Image 2.
- Using the provided evaluation rubrics, determine which image has better overall quality
    .
- Assign a rank value to each image:
- rank = 1 -> higher quality (better image)
- rank = 2 -> lower quality (worse image)
- The output format should be: [rank_of_Image1, rank_of_Image2]
- Examples:
- If Image 1 is better than Image 2 -> output: [1, 2]
- If Image 2 is better than Image 1 -> output: [2, 1]
- Important: The two rank values **must be different** --- ties are not allowed.
- Focus solely on the image content and quality per the rubrics; do **not** let image
    order or
position bias your judgment.
## Output Format
Please strictly follow the JSON format below:
{{
"rank": [rank_of_Image1, rank_of_Image2],
"reason": "Detailed explanation of the reasoning and quality assessment for both images"
}}
Key reminders:
- The ‘rank‘ array must contain exactly 2 values, one for each image.
- rank = 1 means higher quality; rank = 2 means lower quality.
```

Figure 8: **T2I evaluation system prompt.** The prompt template used to instruct the VLM judge to perform pairwise comparison for text-to-image generation, including task description, output format requirements, and anti-position-bias reminders.

Image editing rubrics (Gemini 3.1 Pro)

```
Rubric 1:
Theme: Fidelity of Built Structures After Editing
- Tip 1: Verify that edited building facades, windows, railings, roofs, and other
    structural elements preserve precise geometric accuracy, with
straight lines, correct proportions, and no visible distortions such as bending, warping,
     or duplicated edges.
- Tip 2: Ensure that architectural details, including columns, arches, balconies,
    decorative patterns, and surface textures, remain internally
consistent and logically constructed after editing, without introducing impossible or
    unstable structures.
- Tip 3: Check that material properties of architectural components, such as wood grain,
    stone roughness, brick seams, glass transparency, and
metallic reflections, are faithfully preserved or enhanced without becoming unnaturally
    smooth, plastic-like, or visually artificial.
- Tip 4: Confirm that edits maintain coherent spatial relationships among buildings,
    pathways, roads, vegetation, and surrounding environmental
elements, ensuring perspective consistency and physically plausible scene layout.
Rubric 2:
Theme: Physically Plausible Illumination Adjustments
- Tip 1: Ensure that edited lighting direction, intensity, exposure, and color
    temperature remain physically plausible and align with the
original illumination setup or the newly introduced light source.
- Tip 2: Validate that shadows added, removed, or modified remain globally consistent in
    direction, softness, scale, and occlusion behavior
across all scene objects and surfaces.
- Tip 3: Check that reflections and specular highlights on water, glass, polished metal,
    or glossy surfaces are updated consistently to match
the revised lighting conditions and viewing angle.
- Tip 4: Avoid introducing flat lighting, artificial glow, or inconsistent highlights
    during editing.
Rubric 3:
Theme: Material Integrity During Editing
- Tip 1: Confirm that fine textures (fabric, wood grain, skin, fur) are preserved and not
     blurred or over-smoothed.
- Tip 2: Ensure that material-specific physical properties, including glossiness,
    roughness, transparency, translucency, and reflectivity,
remain realistic and visually consistent after editing.
- Tip 3: Introduce or maintain natural imperfections (scratches, dust, wear) where
    appropriate to avoid artificial perfection.
- Tip 4: Avoid repetitive, tiled, or uniform texture artifacts that may be introduced
    during the editing process, particularly in large
homogeneous regions.
Rubric 4:
Theme: Realism of Subjects and Objects Post-Edit
- Tip 1: Ensure human and animal anatomy remains proportionally accurate after editing (
    no distorted limbs or features).
- Tip 2: Validate that edited objects, such as tools, clothing, furniture, vehicles, or
    accessories, preserve logical structure, correct
functionality, and physical plausibility.
- Tip 3: Detect and correct warped shapes, broken boundaries, or blending artifacts.
- Tip 4: Confirm that subjects interact naturally with their environment (contact,
    shadows, occlusion).
Rubric 5:
Theme: Cohesion and Aesthetic Integrity
- Tip 1: Ensure edits preserve or improve scene balance and visual hierarchy.
- Tip 2: Validate perspective, framing, and depth relationships after modifications.
- Tip 3: Maintain smooth integration between foreground, midground, and background
    elements, avoiding abrupt transitions, cutout artifacts, or
mismatched blur levels.
- Tip 4: Avoid clutter, awkward cropping, or disruption of focal points.
Rubric 6:
Theme: Elimination of AI and Editing Artifacts
- Tip 1: Identify and remove common artifacts such as smearing, halos, pixelation, jagged
     edges, or unnatural blur introduced during editing or
```

Image editing rubrics (Gemini 3.1 Pro) (continued)

```
generation.
- Tip 2: Avoid introducing repeated patterns, duplicated elements, or inconsistent
    geometry.
- Tip 3: Ensure seamless blending between edited and original regions, with smooth
    transitions in texture, lighting, and color.
- Tip 4: Prioritize clean, polished outputs with minimal visible traces of manipulation,
    ensuring the image appears naturally captured rather
than artificially altered.
Rubric 7:
Theme: Background and Context Integrity
- Tip 1: Ensure environmental elements (e.g., trees, sky, terrain, buildings) remain
    physically plausible and visually consistent after editing.
- Tip 2: Maintain consistent scale, depth, and perspective between foreground and
    background.
- Tip 3: Avoid repetitive, tiled, or overly uniform patterns in natural elements, which
    may indicate synthetic generation.
- Tip 4: Ensure all background details align contextually with the subject, supporting a
    coherent scene narrative and avoiding mismatched
environments.
Rubric 8:
Theme: Clarity of Edited Text Elements
- Tip 1: Ensure all text remains clearly legible after editing, with crisp edges and
    consistent, recognizable font styles.
- Tip 2: Preserve correct spelling, grammar, and semantic meaning, avoiding nonsensical
    or AI-generated text artifacts.
- Tip 3: Maintain proper alignment, spacing, and contrast to ensure readability against
    varying backgrounds.
- Tip 4: Avoid distortions such as warping, stretching, or irregular baselines that
    degrade text quality or professionalism.
Rubric 9:
Theme: Harmonized Color Adjustments
- Tip 1: Ensure color palettes remain balanced and contextually appropriate to the scene,
     avoiding abrupt or unrealistic shifts.
- Tip 2: Maintain smooth and natural gradients, preventing banding, clipping, or harsh
    transitions between tones.
- Tip 3: Align color grading with the scene's lighting conditions, time of day, and
    intended mood or atmosphere.
- Tip 4: Avoid oversaturation, color mismatches, or unnatural hues that reduce realism or
     visual comfort.
Rubric 10:
Theme: Micro-Detail Integrity in Editing
- Tip 1: Preserve sharpness and fidelity in fine details such as hair strands, fabric
    stitching, surface engravings, or textures.
- Tip 2: Ensure micro-textures remain natural and are not overly smoothed or artificially
     sharpened during processing.
- Tip 3: Maintain variation and randomness in small-scale details to avoid repetitive or
    algorithmically generated patterns.
- Tip 4: Prevent loss of high-frequency information, ensuring that intricate regions
    retain clarity and do not degrade during editing.
```

Figure 9: **Auto-generated image editing rubrics (Gemini 3.1 Pro).** Example prompt-conditioned rubrics automatically synthesized by ARR for image editing evaluation, covering fidelity preservation, material integrity, lighting consistency, and artifact elimination.

Image editing evaluation system prompt

```
Please perform a pairwise comparison between the two edited versions (**Image 1** and **
    Image 2**)
based on the given rubrics and caption, using the **original image (Image BASE)** as the
reference benchmark. Determine which edited result is superior.
{task_description_section}
**Evaluation Rubrics:**
{rubrics}
**caption: {sample_content}**
## Task Requirements
- You are evaluating two image editing results (Image 1 and Image 2) with **Image BASE (
    the
original image)** as the ground-truth reference.
- Compare them across the following aspects:
- Fidelity to the original content
- Accuracy and naturalness of instruction following
- Assign rank values to the two edited images:
- rank=1 means better quality (higher fidelity to Image BASE + better editing)
- rank=2 means worse quality
- Output format must be: [rank_of_Image1, rank_of_Image2]
- Examples:
- If Image 1 is better -> [1, 2]
- If Image 2 is better -> [2, 1]
- Important: The two ranks must be different (strictly 1 and 2). All judgments must
    explicitly
reference Image BASE.
- Judge solely based on image content, the provided rubrics, and comparison with the
    original
Image BASE. Do not be influenced by image position or order.
## Output Format
Please strictly follow this JSON format:
{
"rank": [rank of Image 1, rank of Image 2],
"reason": "Detailed explanation of the judgment, including comparison of each edited
    image
with the original Image BASE"
}
**Key Reminder:**
- The "rank" array must contain exactly two values corresponding to Image 1 and Image 2.
- rank=1 indicates the better edit (higher fidelity + better execution), rank=2 indicates
     the
worse one.
```

Figure 10: **Image editing evaluation system prompt.** The prompt template used to instruct the VLM judge to perform pairwise comparison for image editing, where Image BASE serves as the ground-truth reference for fidelity assessment.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly enumerate the three core contributions (ARR, RPO, and the interface-bottleneck diagnosis). All quantitative claims (e.g., 1.7–6.3 point accuracy gains; GenEval 0.66→0.80) are directly substantiated by Tables 1–2 and Figure 2. Aspirational framing is appropriately distinguished from demonstrated results.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Appendix F provides a dedicated Limitations section covering three axes: (1) the frozen-model assumption and the potential of fine-tuning for further gains; (2) the pairwise formulation as a deliberate design choice and its scope relative to pointwise/listwise settings; and (3) the gap between fully automated rubric generation and human-curated rubric precision, with human-in-the-loop refinement identified as future work.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The paper does not present formal theorems or lemmas requiring proof. The mathematical content (Equations 1–13) consists of probabilistic model definitions, objective function formulations, and the RPO policy gradient, all of which are algorithmic specifications rather than theoretical claims requiring formal proof.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Appendix A provides a comprehensive account of all experimental details: dataset descriptions and splits (§A.1), evaluation protocols including forward/reverse positional probing (§A.2), ARR instantiation parameters and verification/refinement budget ($T_{\max} = 5$, 87% pass rate) (§A.3), and full RPO hyperparameters including learning rates, batch sizes, LoRA ranks, KL coefficients, and reward magnitudes (§A.4, Table 3). The full RPO algorithm is formalized in Algorithm 1. System prompts and rubric examples are provided in Appendix I.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.

(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Code and reproduction instructions are publicly available at `https://github.com/OpenEnvision/AutoRubric-as-Reward`. All training data are drawn from publicly available sources (ShareGPT-4o-Image). Official benchmark test sets are used only for evaluation. The supplemental material includes prompt templates (Appendix I) and the full algorithmic specification (Algorithm 1) to support reproduction.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Section 4.1 summarizes all baselines and implementation choices. Appendix A.2 details evaluation protocols (sampling steps 30/50, guidance scales 3.5/4.0). Appendix A.4 and Table 3 report all RPO hyperparameters (optimizer: AdamW, gradient clipping: 1.0, clip ϵ: 0.2, LoRA rank: 16/32) and their rationale. Rubric cardinality is fixed at $K = 5$ by default with ablations over $K \in \{1, 5, 10, 20\}$ in Appendix D.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Table 1 reports standard deviation error bars ($\pm$) over three independent evaluation runs for all ARR preference-accuracy results across all four benchmarks (e.g., $\pm$0.1–0.4%). Section 4.2 describes how the mean and standard deviation are computed from the three run-level accuracies. Generative benchmark scores (Table 2) are reported as point estimates.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: RPO experiments ran on a cluster of eight NVIDIA H100 GPUs (80 GB SXM5). For each judge, we ran 300 iterations per task, with 32 rollout prompts per iteration for text-to-image generation and 16 for image editing. Each prompt yielded two candidate images and required one pairwise judge comparison. Across both tasks, this amounts to 14,400 online comparisons and 28,800 candidate images per judge, with an estimated compute cost of 160–240 H100 GPU-hours per judge. The rubric was constructed offline once per judge and task and reused throughout RPO training. Appendix A and Table 3 provide the training configuration.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

Answer: [Yes]

Justification: The research uses publicly released datasets and benchmarks, open-weight models (e.g., FLUX.1-dev, Qwen-Image-Edit-2509, Qwen3-VL-8B, BAGEL), and proprietary models accessed through their official APIs (GPT-5, Gemini 3.1 Pro). All of these are properly cited and used in accordance with their licenses and terms of use. No new human subjects research or crowdsourcing was conducted; all human preference labels come from existing public benchmarks. We did not collect personal data or attempt to identify any individual appearing in the benchmark images.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: On the positive side, ARR-RPO enables more interpretable and bias-resilient alignment of generative models with human preferences, reducing the opacity of reward signals and potentially improving the safety and reliability of deployed image generation systems. On the negative side, improved generative quality could lower barriers to producing synthetic disinformation or non-consensual imagery; the framework's training-free nature makes it particularly accessible, which amplifies both beneficial and potentially harmful uses. We recommend that practitioners deploying ARR-RPO-aligned models implement content filtering and usage monitoring.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: This paper proposes an evaluation and alignment framework applied to existing publicly released base models (FLUX.1-dev, Qwen-Image-Edit-2509). The released artifacts (rubric generation code, LoRA adapters) do not constitute standalone high-risk generative systems; misuse risk is inherited from the underlying base models, which are governed by their respective existing terms of use.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We cite the original work and report the publicly documented licenses below. The HPDv3 dataset card lists MIT, while the authors' published terms (Sec. 9) specify noncommercial use and identify CC BY-NC-SA. The public TIIF-Bench testmini prompt release leaves the prompt license unspecified.

The MM-RewardBench2 release is CC BY-NC 4.0. Its response-image release lists CC BY-NC-SA 4.0, and third-party assets retain their own terms. ShareGPT-4o-Image, EditReward-Bench, and the ImgEdit data release list Apache 2.0. GenEval and GEdit-Bench list MIT. DPG-Bench is distributed in the Apache 2.0 ELLA repository. The UniGenBench++ dataset is CC BY 4.0.

Qwen-Image-Edit-2509 and Qwen3-VL-8B-Instruct list Apache 2.0. FLUX.1-dev is subject to the FLUX.1 [dev] Non-Commercial License. GPT-5 and Gemini 3.1 Pro were accessed through the providers' APIs under the OpenAI and Gemini API terms, respectively.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The public repository provides the ARR rubric generation and verification code, RPO training scripts, and setup, data preparation, and reproduction instructions. The code license is Apache 2.0. Hugging Face hosts the FLUX.1-dev LoRA and the image-editing LoRA, including their adapter configurations. The model card documents training settings, intended uses, and limitations. Its metadata lists Apache 2.0, and the card states that the base-model terms also apply. The card identifies `Qwen/Qwen-Image-Edit` as the base for the published editing adapter. The experimental setup in this paper specifies Qwen-Image-Edit-2509.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This paper does not conduct new crowdsourcing or human subject experiments. All human preference labels used for evaluation are drawn from pre-existing publicly released benchmarks (HPDv3, MM-RewardBench2, EditReward-Bench), whose original collection procedures are described in the respective cited works.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: No new human subject research was conducted. All human preference data used in this work were collected by third parties under their own ethical oversight and are used here solely for evaluation purposes in accordance with their terms of release.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research?

    Answer: [Yes]

    Justification: LLMs and VLMs are a central and original component of the proposed methodology. Specifically, frozen VLMs (Qwen3-VL-8B, GPT-5, Gemini 3.1 Pro) serve as rubric generators (M_{gen}), verifiers (M_{verify}), refiners (M_{refine}), hierarchical structurers (M_{struct}), and preference judges throughout ARR and RPO. Their roles, prompting strategies, and system prompts are fully documented in §3.2, §3.3, Appendix B, and Appendix I.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.