CHECKDISTILL: ADAPTIVE CHECKLIST DISTILLATION FOR EFFICIENT DISTRIBUTIONAL REWARD MODELING IN IMAGE EDITING

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning for instruction-guided image editing requires reward models that provide reliable, accurate, and efficient feedback. Direct scalar prediction compresses multidimensional editing preferences, whereas generative evaluators can expose their judgment through explicit reasoning. However, broad scoring criteria do not specify all instance-dependent editing and preservation constraints, and generating long reasoning trajectories is costly for online optimization. We introduce CheckDistill, a pointwise reward modeling framework that internalizes instruction-adaptive checklists. We decompose each editing condition into atomic, independently verifiable requirements and construct four-dimensional scoring labels together with structured verification evidence. After supervised score initialization, a privileged teacher conditions on checklists, item-level judgments, and dimension-level rationales to supervise the student's own scoring trajectories. A reliability-weighted on-policy self-distillation objective adjusts supervision at score tokens according to agreement between the teacher's expected scores and reference labels, while retaining candidate probability weights in a truncated token support. The deployed student directly predicts four score distributions without checklist inputs or explicit reasoning and aggregates their expectations into a scalar reward. On Qwen3.5-9B, CheckDistill achieves aggregate accuracies of 72.30%, 56.67% and 82.90% on MMRB2, MER-Bench and EditReward-Bench, respectively, exceeding the compared open-source baselines on all three metrics.

1 INTRODUCTION

Instruction-guided image editing is a key application of multi-modal generation, enabling users to perform diverse image modifications through natural language instructions. Reinforcement learning can substantially improve image editing performance, with reward models playing a critical role in guiding optimization. An effective reward model must evaluate candidate results across complementary dimensions, distinguish subtle yet consequential differences, and provide accurate preference rankings and score singals. Frequent reward queries during online reinforcement learning further require reward models to combine comprehensive, fine-grained evaluation with efficient scoring.

Existing reward modeling approaches do not fully meet these requirements. As shown in Fig. 1(a), regression-based reward models (Xu et al., 2023; Wu et al., 2023a; 2026) enable efficient scoring. However, these models rely on coarse-grained subjective preferences, typically learned through pairwise objectives such as the Bradley–Terry loss (Bradley & Terry, 1952). These objectives alone provide neither explicit verification evidence nor absolute score calibration. These limitations reduce interpretability and may make score distributions more prone to drift and the models more vulnerable to reward hacking. Fig. 1(b) shows the reasoning-based generative reward models (Luo et al., 2026; Wang et al., 2026; Kan et al., 2026). This paradigm uses fixed rubric prompts to guide evaluation through explicit reasoning before scoring. However, even when rubrics are generated adaptively, dimension-level prompts may leave concrete verification targets insufficiently specified. The resulting reasoning may not systematically cover the instance-specific editing and preservation constraints that distinguish otherwise similar candidates. Moreover, reasoning-based scoring requires additional autoregressive decoding at each evaluation. Although downstream optimization

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

uses the resulting reward scores, generating the accompanying rationales adds latency and limits throughput during online RL training.

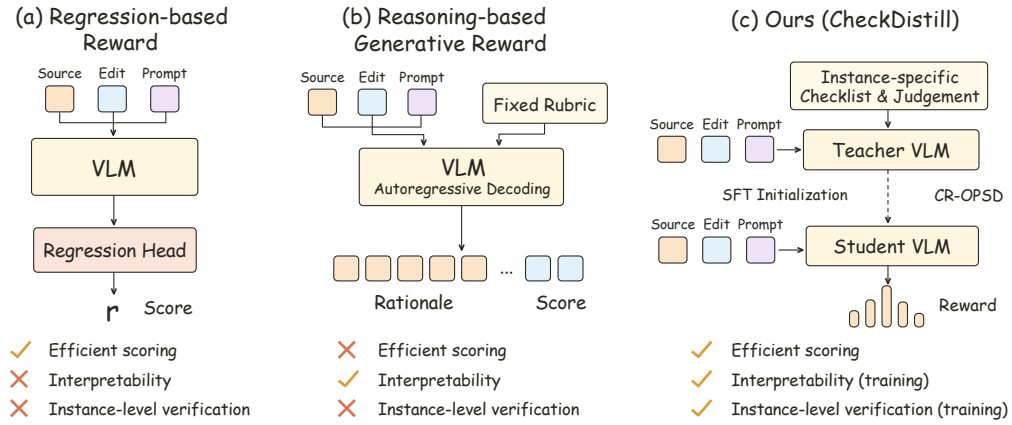


Figure 1: **Overview** of CheckDistill compared with existing reward model architecture.

These limitations motivate two complementary principles. First, **Dynamic Checklist Scaling**: reward judgments should be grounded in instance-adaptive identification of editing and preservation requirements, decomposed into fine-grained, atomic, and independently verifiable checks. Expanding the coverage and specificity of these checks can support more accurate and robust reward judgments. Second, **Context-Compressed Reward Scoring**: efficient online deployment requires reducing the decoding overhead of lengthy chain-of-thought reasoning context while preserving reward quality. Therefore, the core challenge is to align score distributions under standard editing inputs with their evidence-conditioned counterparts, while reducing inference-time dependence on extensive verification contexts and explicit reasoning.

Guided by these principles, we propose **CheckDistill**, an image editing reward modeling framework that combines dynamic checklist-based data construction with reliability-aware on-policy self-distillation, as illustrated in Fig. 1(c). The framework builds structured verification evidence during training and transfers evidence-enhanced judgments into efficient direct scoring.

For data construction, Fig. 2 illustrates how the source image and editing instruction guide atomic checks for Instruction Following (IF), Attribute Preservation (AP), and Context Preservation (CP). These checks are shared across candidates under the same editing condition. In the illustrated example, item-level judgments identify an unchanged like count and an undeleted comment, while separately verifying preservation requirements. Perceptual Quality (PQ) is assessed independently using a fixed general rubric. Combining EditReward-Data (Wu et al., 2026) with additional Qwen-Image-Edit (Wu et al., 2025) candidates, we construct **CheckDistill-40K** through this annotation pipeline and human preference-consistency filtering, pairing four-dimensional score labels with item-level judgments and dimension-level rationales.

CheckDistill adopts two-stage training. Mixed SFT first establishes direct scoring and checklist-guided reasoning capabilities. Then we introduce **Checklist-Guided Reliability-Aware On-Policy Self-Distillation (CR-OPSD)**. The teacher additionally receives checklists, item-level judgments, and dimension-level rationales, providing distributional supervision on response prefixes generated by a student that observes only standard editing inputs. Since evidence-enhanced predictions may still be unreliable, we use agreement between the teacher's expected score and the reference label as a reliability proxy to modulate distillation at the four score positions. At inference time, only the student is retained, and expected scores from its four-dimensional score distributions are aggregated into a scalar reward, without additional verification context or lengthy rationale generation.

CheckDistill achieves aggregate preference accuracies of 72.30% on MMRB2, 56.67% on MER-Bench, and 82.90% on EditReward-Bench, outperforming the compared open-source baselines across all three benchmarks. When used as the reward model for DiffusionNFT (Zheng et al., 2026) post-training of Qwen-Image-Edit, it achieves scores of 4.46 on ImgEdit-Bench, 4.26 on REDEdit-Bench, and 8.28 on GEdit-EN.

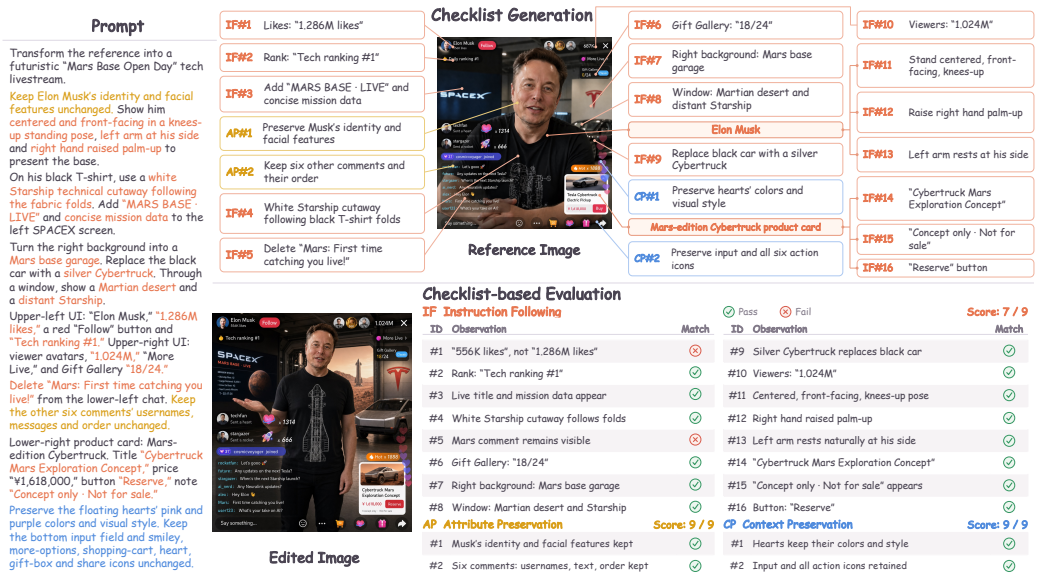


Figure 2: An example of checklist generation and checklist-based evaluation for image editing.

Our main contributions are as follows:

- **Dynamic checklist-based reward supervision.** We construct **CheckDistill-40K**, pairing four-dimensional scores with instance-specific verification evidence through dynamic checklists and human preference-consistency filtering.
- **Reliability-aware distillation for efficient reward scoring.** We propose **CR-OPSD** to transfer evidence-conditioned scoring behavior into a direct reward model through reliability-aware on-policy distribution alignment.
- **Leading reward accuracy and downstream editing gains.** CheckDistill outperforms the compared state-of-the-art open-source reward models on image editing reward benchmarks and improves Qwen-Image-Edit through reward-guided post-training.

2 Related Work

Reinforcement learning (RL) (Black et al., 2024; Fan et al., 2023; Liu et al., 2025a; Xue et al., 2025; Zheng et al., 2026) has emerged as an important post-training paradigm for visual generation, enabling diffusion and flow-based models (Rombach et al., 2022; Podell et al., 2024; Esser et al., 2024; Black Forest Labs et al., 2025) to achieve stronger alignment. Its effectiveness, however, relies heavily on accurate and efficient reward models.

Regression-based Reward Modeling. Regression-based visual reward models learn human preferences from vision-language representations using regression or Bradley–Terry-style preference objectives (Wu et al., 2023b; Xu et al., 2023; Kirstain et al., 2023; Wu et al., 2023a; Zhang et al., 2024b; Wu et al., 2024b; Zhang et al., 2024a; Chen et al., 2025; Ma et al., 2025; Wang et al., 2025b; Wu et al., 2026). These methods are efficient at inference, but typically output scalar scores without explicit reasoning, limiting interpretability and diagnostic feedback for fine-grained evaluation.

Generative Reward Modeling. Generative reward models such as EditScore (Luo et al., 2026), RationalRewards (Wang et al., 2026), SpatialReward (Long et al., 2026), and UnifiedReward-Think (Wang et al., 2025a) produce explicit rationales or chain-of-thought reasoning for fine-grained reward prediction under fixed evaluation criteria. However, these criteria are not instantiated into concrete, input-specific verification requirements, and inference-time reasoning introduces substantial overhead in large-scale online RL. JRM (Yang et al., 2026) and Z-Reward (Jin et al., 2026)

improve efficiency by internalizing reasoning into direct reward prediction, but still lack instance-adaptive verification criteria.

Instance-adaptive methods tailor evaluation criteria to each input. RubricRM (Kan et al., 2026) generates input-specific dimensions and scoring criteria, but remains focused on high-level evaluation dimensions. Edit-R1 (Guo et al., 2026) decomposes editing instructions into sample-specific principles for finer-grained verification, but still requires explicit reasoning at inference time. Checklist-based methods further translate evaluation requirements into directly verifiable items (Cook et al., 2024); rDPO (Yu et al., 2026), for example, constructs instance-specific checklist-style rubrics primarily for preference data construction. In contrast, CheckDistill uses source-image-grounded atomic checklists only as privileged information during reward-model training, distilling fine-grained instance-specific verification requirements into direct multidimensional reward prediction without checklist generation or explicit reasoning at inference time.

3 METHOD

As shown in Figure 3, we propose CheckDistill, a generative reward model that leverages instance-specific checklists for fine-grained evaluation and internalizes checklist-derived evaluation knowledge through CR-OPSD. Given an input $x = (u, I^{\mathrm{src}}, I^{\mathrm{edt}})$, CheckDistill maps it to a multi-dimensional scoring response y, where u denotes the editing instruction, I^{src} the source image, and I^{edt} the edited image.

3.1 EVALUATION FORMULATION

Evaluation Dimensions. We evaluate image editing from two complementary aspects: *Semantic Consistency* (SC) and *Perceptual Quality* (PQ). SC comprises three dimensions: *Instruction Following* (IF), which measures whether the requested edits are correctly executed; *Attribute Preservation* (AP), which measures whether unspecified attributes of edited objects are preserved; and *Context Preservation* (CP), which measures whether non-edited content remains unchanged. PQ measures the overall visual quality of the edited image. All four scores, S_{IF}, S_{AP}, S_{CP}, and S_{PQ}, are defined on a discrete scale from 0 to 9.

Multidimensional Reward Aggregation. To obtain a scalar feedback signal for downstream post-training of the editing model, we account for the varying importance of different dimensions by first aggregating the three SC scores using dimension-specific weights, and then combining SC and PQ through a weighted geometric mean:

$$R_{\mathrm{SC}}(x) = \sum_{\delta \in \mathcal{D}_{\mathrm{SC}}} w_\delta S_\delta(x), \qquad R(x) = R_{\mathrm{SC}}(x)^{\lambda} S_{\mathrm{PQ}}(x)^{1-\lambda}. \tag{1}$$

where $\sum_{\delta \in \mathcal{D}_{\mathrm{SC}}} w_\delta = 1$ and $w_\delta \geq 0$, with $\mathcal{D}_{\mathrm{SC}} = \{\mathrm{IF}, \mathrm{AP}, \mathrm{CP}\}$, and $\lambda \in [0, 1]$ controls the relative contribution of semantic consistency and perceptual quality.

3.2 REWARD DATA CONSTRUCTION PIPELINE

To construct the reward training data, we develop an annotation pipeline that leverages instance-specific checklists to facilitate fine-grained evaluation and reliable score annotation, while also providing structured privileged information for subsequent knowledge internalization via CR-OPSD.

Candidate Sampling and Human Preferences. Given an editing condition $h = (u, I^{\mathrm{src}})$, we generate multiple edited images with an image editing model using different random seeds, forming a diverse candidate pool $\mathcal{I}^{\mathrm{edt}} = \{I_k^{\mathrm{edt}}\}_{k=1}^{K}$. Human annotators then perform pairwise comparisons among the candidates. For each annotated pair, the preferred and non-preferred candidates are denoted as I_w^{edt} and I_l^{edt}, respectively, yielding the human preference relation $I_w^{\mathrm{edt}} \succ I_l^{\mathrm{edt}}$.

Hybrid Rubric Design. Our hybrid rubric consists of an instance-specific checklist-based rubric for SC and a fixed general rubric for PQ, with the detailed rubric templates provided in Appendix D.

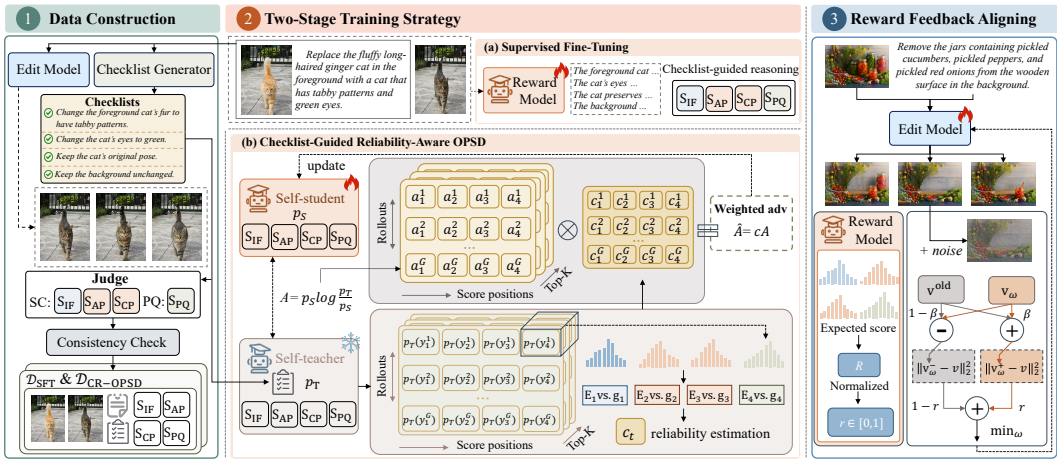


Figure 3: **Overview of CheckDistill**, comprising reward data construction, two-stage reward model training via supervised fine-tuning and checklist-guided reliability-aware OPSD, and reward feedback alignment for image editing.

(1) Instance-specific checklist for SC. To provide more fine-grained and traceable evidence than direct dimension-level assessment, we decompose the three SC dimensions into an instance-specific checklist of atomic verification items, each grounded in concrete objects, attributes, or regions. Specifically, for each editing condition h, we use Gemini-3.0-Flash to generate a variable-length checklist $C = \{c_j\}_{j=1}^{M}$, without access to any candidate result. where M denotes the number of checklist items. The same checklist C is shared across all candidates under the same editing condition, ensuring that they are evaluated against a consistent set of criteria.

(2) General rubric for PQ. Since PQ concerns generic visual quality rather than instance-specific editing constraints, we adopt a fixed rubric shared across all samples and assess PQ directly from the edited image without reference to the source image. The rubric focuses on key perceptual factors such as realism and overall naturalness.

Independent SC and PQ Judgment. We employ two independent Gemini-3.0-Flash annotation templates (see Appendix D) for SC and PQ assessment. For the k-th candidate, the SC annotator takes (x_k, C) as input and verifies each checklist item according to its associated dimension, producing item-level judgments J_k. These judgments are then summarized into dimension-level rationales A_k^{SC} and scores $\mathbf{g}_k^{\mathrm{SC}} = (g_k^{\mathrm{IF}}, g_k^{\mathrm{AP}}, g_k^{\mathrm{CP}})$. The PQ annotator assesses the visual quality of the edited result from (u, I_k^{edt}) and outputs a rationale A_k^{PQ} and score g_k^{PQ}.

Preference-Consistency Filtering. We use the human preferences collected in Sec. 3.2 as an external reference to verify the ranking consistency of the automatically annotated labels. We perform filtering at the candidate-group level. For each annotated pair with $I_w^{\mathrm{edt}} \succ I_l^{\mathrm{edt}}$, we aggregate the four-dimensional scores according to Eq. 1 and regard the pair as preference-consistent when $R_w > R_l$. Since each candidate may participate in multiple pairwise comparisons, we retain the largest subset whose induced ranking agrees with all available human preferences among the retained candidates.

Through the above pipeline, we construct the **CheckDistill-40K** dataset, comprising approximately 20K samples from EditReward-Data (Wu et al., 2026) and 20K samples generated with Qwen-Image-Edit following Sec. 3.2. For the first-stage SFT, we use a 10K subset, denoted as $\mathcal{D}_{\mathrm{SFT}}$, and mix two supervision formats: (1) given x_k as input, the model predicts the four-dimensional scores $\mathbf{g}_k = [g_k^{\mathrm{IF}}, g_k^{\mathrm{AP}}, g_k^{\mathrm{CP}}, g_k^{\mathrm{PQ}}]$; and (2) given (x_k, C) as input, the model is supervised to generate the checklist judgments J_k, rationales $A_k = (A_k^{\mathrm{SC}}, A_k^{\mathrm{PQ}})$, and scores $\mathbf{g}_k$. For the second-stage CR-OPSD training, we use the full dataset $\mathcal{D}_{\text{CR-OPSD}} = \{(x_k, \mathbf{g}_k, z_k)\}_{k=1}^{N}$, where $z_k = (C, J_k, A_k)$ denotes the structured privileged information.

3.3 CHECKDISTILL TRAINING AND DOWNSTREAM ALIGNMENT

Stage I: Supervised Initialization. We first perform supervised fine-tuning to initialize CheckDistill with checklist-guided reasoning and the ability to directly predict four-dimensional scores from the editing instruction, source image, and edited result. Let $p_\theta(y \mid x')$ denote the conditional autoregressive distribution over responses defined by CheckDistill with parameters θ, where x' denotes the input for the corresponding training format. The training objective is

$$\mathcal{L}_{\mathrm{SFT}}(\theta) = -\mathbb{E}_{(x',y)\sim\mathcal{D}_{\mathrm{SFT}}}\left[\sum_{t=1}^{|y|}\log p_\theta(y_t \mid x', y_{<t})\right]. \tag{2}$$

where $(x', y) \in \{(x, \mathbf{g}),\ ((x, C), (J, A, \mathbf{g}))\}$. This stage elicits checklist-guided reasoning and establishes the score range and output structure, providing a foundation for CR-OPSD.

Stage II: Checklist-Guided Reliability-Aware OPSD (CR-OPSD). We then propose CR-OPSD to internalize fine-grained checklist-guided reasoning into the reward model for efficient inference. Specifically, CR-OPSD builds on OPSD (Zhao et al., 2026) by using the structured evidence $z = (C, J, A)$ as privileged information and introducing reliability-aware weighting to reduce the influence of inaccurate teacher predictions. Both the student and teacher are initialized from the SFT checkpoint. Their predictive distributions are defined as

$$p_S(y_t \mid x, y_{<t}) = p_\theta(y_t \mid x, y_{<t}), \qquad p_T(y_t \mid x, z, y_{<t}) = p_\theta(y_t \mid x, z, y_{<t}). \tag{3}$$

Let $\mathcal{T}_t = \mathrm{TopK}(p_S(\cdot \mid x, y_{<t}))$ denote the student's Top-K tokens at position t. We define $\mathcal{S} = \{0, \ldots, 9\}$ as the set of valid score tokens and $\mathcal{S}_t = \mathcal{S} \cap \mathcal{T}_t$. At each score position, we adopt a Q-Align-style (Wu et al., 2024a) probabilistic readout over $\mathcal{S}_t$ to compute the teacher's expected score:

$$\bar{p}_T(s \mid x, z, y_{<t}) = \frac{p_T(s \mid x, z, y_{<t})}{\sum_{s'\in\mathcal{S}_t} p_T(s' \mid x, z, y_{<t})}, \qquad E_t = \sum_{s\in\mathcal{S}_t} s\,\bar{p}_T(s \mid x, z, y_{<t}). \tag{4}$$

Let $\delta(t)$ denote the scoring dimension associated with position t, with annotated score $g^{\delta(t)}$. We measure teacher–label agreement as

$$r_{\mathrm{unit},t} = \mathrm{clip}\left(1 - \frac{|E_t - g^{\delta(t)}|}{9}, 0, 1\right), \qquad r_t = r_{\min} + (r_{\max} - r_{\min})\frac{1 - \cos(\pi r_{\mathrm{unit},t})}{2}. \tag{5}$$

The position-wise coefficient is defined as $c_t = (1 - m_t) + m_t r_t$, where $m_t = 1$ at the four score positions and $m_t = 0$ otherwise. For each $y_t \in \mathcal{T}_t$, the reliability-weighted advantage is

$$\hat{A}_t(y_t) = \mathrm{sg}\left[c_t\, p_S(y_t \mid x, y_{<t}) \log \frac{p_T(y_t \mid x, z, y_{<t})}{p_S(y_t \mid x, y_{<t})}\right], \qquad y_t \in \mathcal{T}_t. \tag{6}$$

where $\mathrm{sg}[\cdot]$ denotes the stop-gradient operator. Aggregating over the Top-K candidates at each position gives the final objective:

$$\mathcal{L}_{\mathrm{CR-OPSD}}(\theta) = -\mathbb{E}\left[\frac{1}{|y|}\sum_t \sum_{y_t\in\mathcal{T}_t} \hat{A}_t(y_t)\log p_S(y_t \mid x, y_{<t})\right]. \tag{7}$$

Reward Feedback for Alignment. We use CheckDistill to guide RL post-training of the image editing model with DiffusionNFT (Zheng et al., 2026). For policy optimization, the reward of each candidate is computed from CheckDistill's predicted score distributions. Specifically, rather than directly taking the most likely discrete score token, we derive a continuous score for each dimension from the predicted distribution. For $\delta \in \mathcal{D}_{\mathrm{SC}} \cup \{\mathrm{PQ}\}$, let t_δ denote its corresponding score position. The reward model probabilities over valid score tokens are renormalized to compute the expected score:

$$\bar{p}_S(s \mid x, y_{<t_\delta}) = \frac{p_S(s \mid x, y_{<t_\delta})}{\sum_{s'\in\mathcal{S}} p_S(s' \mid x, y_{<t_\delta})}, \qquad \bar{S}_\delta(x) = \sum_{s\in\mathcal{S}} s\,\bar{p}_S(s \mid x, y_{<t_\delta}), \quad s \in \mathcal{S}. \tag{8}$$

This distribution-based readout preserves uncertainty over the ordered score space, enabling finer-grained reward differentiation even among candidates with the same most likely score token. The resulting expected dimensional scores are then aggregated into the final reward using Eq. 1, which serves as the reward signal for DiffusionNFT-based policy optimization.

Table 1: Reward benchmark accuracy (%, ↑). Submetric results for EditReward, EditScore-8B, and SpatialReward are reproduced from SpatialReward (Long et al., 2026). Bold marks the best result among the compared open-source models in each column.

	MMRB2			MER-Bench				EditReward-Bench		
Model	**Single**	**Multi**	**Overall**	**2 pair**	**3 pair**	**4 pair**	**Overall**	**PF**	**CS**	**Overall**
GPT-4.1	56.33	57.87	56.60	79.50	42.50	18.50	46.83	70.00	65.20	73.40
Gemini-3.1-Pro	65.20	60.11	64.29	79.00	60.50	24.00	54.50	66.30	75.70	79.80
Gemini-3.0-Flash	66.71	58.64	65.34	82.50	63.00	26.00	57.17	68.70	77.90	80.90
Qwen3-VL-8B-Instruct	63.02	57.87	62.10	73.00	42.00	16.00	43.67	77.20	70.90	75.80
Qwen3.5-4B	57.79	54.49	57.20	75.50	36.00	15.50	42.33	74.40	69.60	70.50
Qwen3.5-9B	63.14	58.99	62.40	76.00	46.50	19.00	47.17	83.10	75.60	78.20
EditReward	67.20	59.00	65.70	70.00	49.00	15.50	44.83	83.20	-	79.20
EditScore-8B	57.90	52.80	57.00	61.00	34.00	10.00	35.00	60.80	59.40	69.00
SpatialReward	67.10	60.80	66.10	78.00	49.50	21.50	49.67	68.30	67.20	80.30
GRPO (Qwen3.5-9B)	69.59	**70.22**	69.70	81.50	58.50	**29.00**	56.33	85.40	85.50	81.10
Token-level GRPO (Qwen3.5-9B)	70.19	69.10	70.00	80.50	59.00	27.00	55.50	85.30	84.30	81.70
CheckDistill (Qwen3-VL-8B)	69.22	60.67	67.70	**83.00**	50.50	25.00	52.83	**85.50**	81.20	78.90
CheckDistill (Qwen3.5-4B)	**73.48**	68.54	**72.60**	82.00	53.50	20.00	51.83	84.50	**85.60**	82.70
CheckDistill (Qwen3.5-9B)	73.36	67.42	72.30	82.00	**60.50**	27.50	**56.67**	**85.50**	84.10	**82.90**

4 EXPERIMENTS

4.1 PERFORMANCE ON REWARD MODELING BENCHMARKS

Implementation Details. We implement CheckDistill on Qwen3-VL-8B-Instruct (Bai et al., 2025), Qwen3.5-4B, and Qwen3.5-9B. Unless otherwise specified, we set $K = 10$ and use reliability weights in $[0.8, 1.2]$. For reward aggregation, the IF/AP/CP weights are set to $(0.4, 0.35, 0.25)$, with SC/PQ exponents of $0.8/0.2$. Implementation details are provided in Appendix A.

Baselines and Evaluation Protocol. We evaluate the reward modeling performance on MMRB2 (Hu et al., 2026), MER-Bench (Long et al., 2026), and EditReward-Bench (Luo et al., 2026). We compare CheckDistill with closed-source visual judges (GPT-4.1, Gemini-3.1-Pro, and Gemini-3.0-Flash), untrained reward-model backbones, and specialized reward models including EditReward (Wu et al., 2026), EditScore-8B (Luo et al., 2026), and SpatialReward (Long et al., 2026). We also conduct a controlled comparison on Qwen3.5-9B against GRPO (Shao et al., 2024) and Token-level GRPO under the same experimental settings. More details on the evaluation protocol are provided in Appendix A.4.

Comparison with Baselines. As shown in Table 1, CheckDistill consistently improves over the corresponding backbone models and achieves strong performance across all three benchmarks. In particular, CheckDistill (Qwen3.5-9B) reaches 72.30% on MMRB2, 56.67% on MER-Bench, and 82.90% on EditReward-Bench, outperforming SpatialReward by 6.20, 7.00, and 2.60 percentage points, respectively. It also exceeds GPT-4.1 and Gemini-3.1-Pro on all three aggregate metrics, while outperforming Gemini-3.0-Flash on MMRB2 and EditReward-Bench and remaining competitive on MER-Bench (56.67% vs. 57.17%). The gains are consistent across model scales: the 4B variant achieves the highest MMRB2 overall accuracy of 72.60%, improving its backbone by 15.40 percentage points. Under the same Qwen3.5-9B backbone and experimental settings, CheckDistill also outperforms GRPO and Token-level GRPO (see Appendix B for implementation details), improving MMRB2 overall accuracy by 2.60 and 2.30 percentage points and EditReward-Bench overall accuracy by 1.80 and 1.20 percentage points, respectively. These results demonstrate the effectiveness of CR-OPSD across different backbone families, model scales, and training strategies.

4.2 APPLICATION IN ONLINE RL

Experimental Setup. We conduct online RL experiments on Qwen-Image-Edit (Wu et al., 2025) using DiffusionNFT (Zheng et al., 2026), with training samples drawn from CheckDistill-40K.

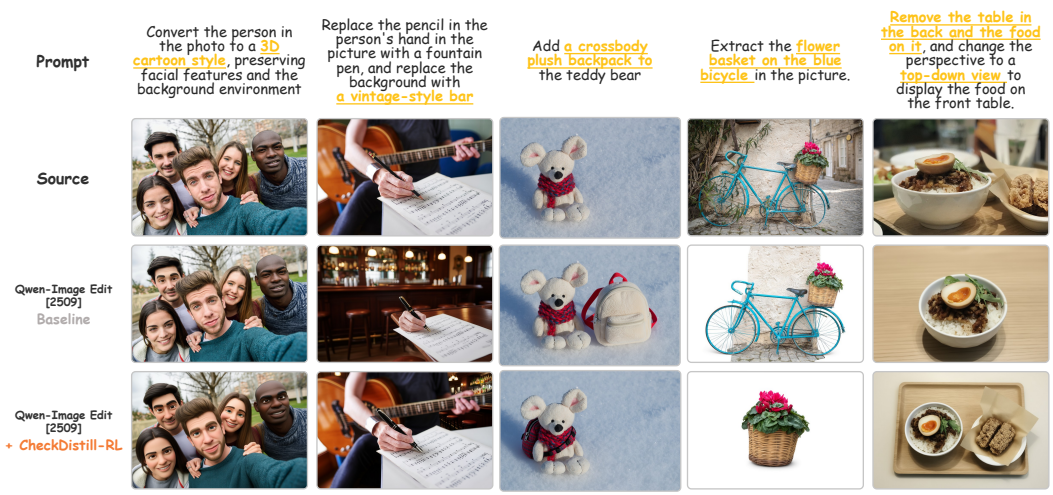


Figure 4: Qualitative Comparison Before and After CheckDistill-based Reward Training.

We evaluate the resulting model on ImgEdit-Bench (Ye et al., 2025), REDEdit-Bench (Super Intelligence Team, 2026), and the English subset of GEdit-Bench (GEdit-EN) (Liu et al., 2025b). We compare CheckDistill-guided RL against the original Qwen-Image-Edit and the open-source UniWorld-Qwen-Image-Edit model (Li et al., 2025).

Editing Performance. As shown in Table 2, RL guided by CheckDistill consistently improves Qwen-Image-Edit across all three benchmarks, increasing ImgEdit-Bench from 4.29 to 4.46, REDEdit-Bench from 4.10 to 4.26, and GEdit-EN from 8.18 to 8.28. It also outperforms UniWorld-Qwen-Image-Edit, improving the three benchmarks from 4.30, 4.06, and 8.12 to 4.46, 4.26, and 8.28, respectively. Qualitative comparisons in Figure 4 further show that CheckDistill-guided RL better follows editing instructions while maintaining stronger consistency with the source image. These results demonstrate the effectiveness of CheckDistill as a reward model for improving downstream image-editing performance through online RL.

Table 2: Image-editing performance (↑).

Model	ImgEdit Bench	REDEdit Bench	GEdit-EN
Qwen-Image-Edit	4.29	4.10	8.18
UniWorld-Qwen-Image-Edit	4.30	4.06	8.12
+ DiffusionNFT (CheckDistill-9B)	**4.46**	**4.26**	**8.28**

4.3 Ablation Study and Analysis

Analysis of Verification Scaling and Efficient Scoring. We evaluate input and output scaling on MMRB2 using Qwen3.5-9B, measuring accuracy and output token counts. In Figure 5, SFT denotes supervised fine-tuning on CheckDistill-40K, while zero-shot refers to evaluation without task-specific fine-tuning.

Instance-specific verification evidence improves reward judgement through input and output scaling. Supplying a checklist improves zero-shot direct-scoring accuracy from 55.76% to 58.90%, while adding checklist-guided judgements raises it to 70.60%, with about 10 output tokens. Generating evaluation content also helps: a generic rationale achieves 61.07%, generating a checklist and its guided judgement reaches 62.40%, and supplying the checklist before generating the judgement achieves 63.41%. These gains support grounding reward as-

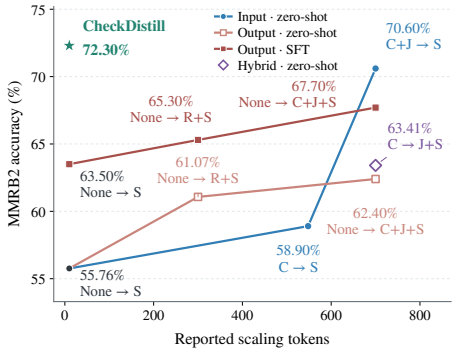


Figure 5: Input, output, and hybrid scaling on MMRB2 (Qwen3.5-9B). Labels show added context → output. C: checklist; J: checklist-guided judgement; R: rationale; S: score; "None": no added context.

Table 3: Ablation studies on privileged information and CR-OPSD design choices (%, ↑).

(a) Privileged-information design

Privileged Information	MMRB2	MER-Bench	EditReward-Bench
Checklist C	72.10	54.83	82.10
+ Checklist Judgment J	72.00	56.00	82.20
+ Dimension-wise Rationale A	**72.30**	**56.67**	**82.90**
+ Dimension Scores $\mathbf{g}$	70.10	55.83	81.30

(b) CR-OPSD design choices

Variant	MMRB2	MER-Bench	EditReward-Bench
w/o Reliability Weighting	70.00	55.00	82.10
Linear Reliability Mapping	71.30	56.17	82.30
Argmax-based Reliability	70.60	55.50	82.60
CR-OPSD (Ours)	**72.30**	**56.67**	**82.90**

sessment in instance-specific verification requirements. In particular, **the strong input-scaling result demonstrates the value of evidence-enriched teacher contexts**, motivating Dynamic Checklist Scaling in CheckDistill.

CheckDistill retains the benefits of evidence-supported evaluation in a compact, direct-scoring model. SFT achieves 63.50% accuracy with direct scoring and 67.70% when generating a checklist and judgement. CheckDistill reaches 72.30% with about 10 output tokens, exceeding these configurations by 8.80 and 4.60 percentage points and surpassing the 70.60% zero-shot result with supplied verification evidence. This supports Distribution-Aligned Context Compression: transferring evidence-enhanced scoring to a student requiring neither verification contexts nor lengthy reasoning at inference time. Compared with the approximately 700-token SFT reasoning configuration, **CheckDistill achieves higher accuracy with approximately 1/70 of the output tokens**, demonstrating a favorable accuracy–decoding-cost trade-off.

Privileged-Information Design. We ablate the privileged information provided to the teacher by progressively introducing checklist C, checklist judgments (C, J), dimension-wise rationales (C, J, A), and dimension scores $(C, J, A, \mathbf{g})$, while keeping the SFT initialization, training data, student inputs, and optimization settings unchanged. As shown in Table 3, (C, J, A) achieves the best overall performance across all three benchmarks, reaching 72.30% on MMRB2, 56.67% on MER-Bench, and 82.90% on EditReward-Bench.

However, further exposing dimension scores to the teacher, i.e., directly providing the ground-truth targets as in OPSD, does not yield additional gains: (C, J, A) outperforms $(C, J, A, \mathbf{g})$ by 2.20, 0.84, and 1.60 points on MMRB2, MER-Bench, and EditReward-Bench, respectively. These results demonstrate the effectiveness of checklist-based privileged information, where checklist judgments and dimension-wise rationales further enrich the supervision without directly exposing score labels.

CR-OPSD Component Ablation. We ablate the key design choices of CR-OPSD while keeping the privileged information (C, J, A), initialization, training data, and optimization settings fixed. Specifically, *w/o Reliability Weighting* removes reliability modulation by setting $c_t = 1$ for all positions; *Linear Reliability Mapping* replaces the cosine mapping in Eq. 5 with a linear mapping; and *Argmax-based Reliability* replaces the expected teacher score with the teacher's maximum-probability score token for reliability estimation.

As shown in Table 3, CR-OPSD consistently achieves the best overall performance across all three benchmarks. Compared with removing reliability weighting, it improves MMRB2, MER-Bench, and EditReward-Bench by 2.30, 1.67, and 0.80 percentage points, respectively, demonstrating the benefit of modulating privileged supervision according to teacher reliability. The cosine mapping further improves over its linear counterpart by 1.00, 0.50, and 0.60 points, while using the expected teacher score instead of the argmax score yields gains of 1.70, 1.17, and 0.30 points. These results validate both reliability-aware weighting and the use of distributional score information for more robust privileged on-policy self-distillation.

5 Conclusion

We present CheckDistill, an efficient reward modeling framework for instruction-guided image editing. It uses instance-specific dynamic checklists for fine-grained verification and CR-OPSD to distill evidence-supported scoring behavior into a direct-scoring model without requiring checklists or explicit reasoning at inference time. Experiments demonstrate strong reward evaluation performance and effective downstream improvements in DiffusionNFT-based image editing.

AI USE STATEMENT

In this work, we used generative AI tools as part of the reward-data construction pipeline to generate structured annotations. We also used generative AI tools to assist with literature search. All generated annotations were processed following the data construction procedures described in the paper, and literature identified with AI assistance was reviewed against the original sources. We take responsibility for the final content of this work, including text, claims, code, data, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We have taken extensive measures to ensure the reproducibility of our results. The main paper provides detailed descriptions of the proposed method, reward-data construction pipeline, training objectives, and evaluation protocols. Additional implementation details, hyperparameter settings, and experimental configurations are provided in the Appendix to facilitate replication of our experiments. We will release the dataset, model checkpoints, and source code to support further verification and reproducibility. Together, these resources provide the necessary information to reproduce our main results and build upon our work.

REFERENCES

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-VL technical report. *arXiv preprint arXiv:2511.21631*, 2025. URL https://arxiv.org/abs/2511.21631.

Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2305.13301.

Black Forest Labs, Stephen Batifol, Andreas Blattmann, Frederic Boesel, Saksham Consul, Cyril Diagne, Tim Dockhorn, Jack English, Zion English, Patrick Esser, Sumith Kulal, Kyle Lacey, Yam Levi, Cheng Li, Dominik Lorenz, Jonas Müller, Dustin Podell, Robin Rombach, Harry Saini, Axel Sauer, and Luke Smith. FLUX.1 Kontext: Flow matching for in-context image generation and editing in latent space. *arXiv preprint arXiv:2506.15742*, 2025. URL https://arxiv.org/abs/2506.15742.

Ralph Allan Bradley and Milton E. Terry. Rank analysis of incomplete block designs: The method of paired comparisons. *Biometrika*, 39(3-4):324–345, 1952. doi: 10.1093/biomet/39.3-4.324. URL https://doi.org/10.1093/biomet/39.3-4.324.

Sherry X. Chen, Yi Wei, Luowei Zhou, and Suren Kumar. ADIEE: Automatic dataset creation and scorer for instruction-guided image editing evaluation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 18345–18356, 2025. URL https://arxiv.org/abs/2507.07317.

Jonathan Cook, Tim Rocktäschel, Jakob Foerster, Dennis Aumiller, and Alex Wang. TICKing all the boxes: Generated checklists improve LLM evaluation and generation. *arXiv preprint arXiv:2410.03608*, 2024. URL https://arxiv.org/abs/2410.03608.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis. In *Proceedings of the 41st International Conference on Machine Learning*, volume

235 of *Proceedings of Machine Learning Research*, pp. 12606–12633. PMLR, 2024. URL https://proceedings.mlr.press/v235/esser24a.html.

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. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/fc65fab891d83433bd3c8d966edde311-Abstract-Conference.html.

Hanzhong Guo, Jie Wu, Jie Liu, Yu Gao, Zilyu Ye, Linxiao Yuan, Xionghui Wang, Yizhou Yu, and Weilin Huang. Leveraging verifier-based reinforcement learning in image editing. *arXiv preprint arXiv:2604.27505*, 2026. URL https://arxiv.org/abs/2604.27505.

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 (CVPR)*, pp. 36904–36915, 2026. URL https://openaccess.thecvf.com/content/CVPR2026/html/Hu_Multimodal_RewardBench_2_Evaluating_Omni_Reward_Models_for_Interleaved_Text_CVPR_2026_paper.html.

Xin Jin, Huanqia Cai, Zhen Li, Zechao Zhan, Dengyang Jiang, Aiming Hao, Yuming Jiang, Xiangpeng Yang, Chunle Guo, Peng Gao, Ming-Ming Cheng, and Steven C. H. Hoi. Z-Reward: Beyond scalar rewards by internalizing reasoning into score distributions. *arXiv preprint arXiv:2606.09076*, 2026. URL https://arxiv.org/abs/2606.09076.

Zijian Kan, Wei Wang, Long Luo, Bing Zhao, Xuan Ren, Weixu Qiao, Wenbo Li, Hu Wei, and Lin Qu. RubricRM: Generative reward modeling via dynamic rubrics for image generation and editing. *arXiv preprint arXiv:2608.26956*, 2026. URL https://arxiv.org/abs/2608.26956.

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. In *Advances in Neural Information Processing Systems*, volume 36, pp. 36652–36663, 2023. doi: 10.52202/075280-1594. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/73aacd8b3b05b4b503d58310b523553c-Abstract-Conference.html.

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:2510.16888*, 2025. URL https://arxiv.org/abs/2510.16888.

Jie Liu, Gongye Liu, Jiajun Liang, Yangguang Li, Jiaheng Liu, Xintao Wang, Pengfei Wan, Di Zhang, and Wanli Ouyang. Flow-GRPO: Training flow matching models via online RL. *arXiv preprint arXiv:2505.05470*, 2025a. URL https://arxiv.org/abs/2505.05470.

Shiyu Liu, Yucheng Han, Peng Xing, Fukun Yin, Rui Wang, Wei Cheng, Jiaqi Liao, Yingming Wang, Honghao Fu, Chunrui Han, Guopeng Li, Yuang Peng, Quan Sun, Jingwei Wu, Yan Cai, Zheng Ge, Ranchen Ming, Lei Xia, Xianfang Zeng, Yibo Zhu, Binxing Jiao, Xiangyu Zhang, Gang Yu, and Daxin Jiang. Step1X-Edit: A practical framework for general image editing. *arXiv preprint arXiv:2504.17761*, 2025b. URL https://arxiv.org/abs/2504.17761.

Yancheng Long, Yankai Yang, Hongyang Wei, Wei Chen, Tianke Zhang, Haonan Fan, Changyi Liu, Kaiyu Jiang, Jiankang Chen, Kaiyu Tang, Bin Wen, Fan Yang, Tingting Gao, Han Li, and Shuo Yang. SpatialReward: Bridging the perception gap in online RL for image editing via explicit spatial reasoning. *arXiv preprint arXiv:2602.07458*, 2026. URL https://arxiv.org/abs/2602.07458.

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 *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/381ceeae4a1feb1abc59c773f7e61839-Abstract-Conference.html.

Yuhang Ma, Yunhao Shui, Xiaoshi Wu, Keqiang Sun, and Hongsheng Li. HPSv3: Towards wide-spectrum human preference score. *arXiv preprint arXiv:2508.03789*, 2025. URL https://arxiv.org/abs/2508.03789.

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 *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/081b08068e4733ae3e7ad019fe8d172f-Abstract-Conference.html.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10684–10695, 2022. URL https://openaccess.thecvf.com/content/CVPR2022/html/Rombach_High-Resolution_Image_Synthesis_With_Latent_Diffusion_Models_CVPR_2022_paper.html.

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

Super Intelligence Team. FireRed-Image-Edit-1.0 technical report. *arXiv preprint arXiv:2602.13344*, 2026. URL https://arxiv.org/abs/2602.13344.

Haozhe Wang, Cong Wei, Weiming Ren, Jiaming Liu, Fangzhen Lin, and Wenhu Chen. RationalRewards: Reasoning rewards scale visual generation both training and test time. *arXiv preprint arXiv:2604.11626*, 2026. URL https://arxiv.org/abs/2604.11626.

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. *arXiv preprint arXiv:2505.03318*, 2025a. URL https://arxiv.org/abs/2505.03318.

Yibin Wang, Yuhang Zang, Hao Li, Cheng Jin, and Jiaqi Wang. Unified reward model for multimodal understanding and generation. *arXiv preprint arXiv:2503.05236*, 2025b. URL https://arxiv.org/abs/2503.05236.

Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng-ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, Yuxiang Chen, Zecheng Tang, Zekai Zhang, Zhengyi Wang, An Yang, Bowen Yu, Chen Cheng, Dayiheng Liu, Deqing Li, Hang Zhang, Hao Meng, Hu Wei, Jingyuan Ni, Kai Chen, Kuan Cao, Liang Peng, Lin Qu, Minggang Wu, Peng Wang, Shuting Yu, Tingkun Wen, Wensen Feng, Xiaoxiao Xu, Yi Wang, Yichang Zhang, Yongqiang Zhu, Yujia Wu, Yuxuan Cai, and Zenan Liu. Qwen-Image technical report. *arXiv preprint arXiv:2508.02324*, 2025. URL https://arxiv.org/abs/2508.02324.

Haoning Wu, Zicheng Zhang, Weixia Zhang, Chaofeng Chen, Liang Liao, Chunyi Li, Yixuan Gao, Annan Wang, Erli Zhang, Wenxiu Sun, Qiong Yan, Xiongkuo Min, Guangtao Zhai, and Weisi Lin. Q-Align: Teaching LMMs for visual scoring via discrete text-defined levels. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 54015–54029. PMLR, 2024a. URL https://proceedings.mlr.press/v235/wu24ah.html.

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 *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/4d68af3056842229f8b356d686aec914-Abstract-Conference.html.

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*, 2023a. URL https://arxiv.org/abs/2306.09341.

Xiaoshi Wu, Keqiang Sun, Feng Zhu, Rui Zhao, and Hongsheng Li. Human preference score: Better aligning text-to-image models with human preference. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 2096–2105, 2023b. URL https://openaccess.thecvf.com/content/ICCV2023/html/Wu_Human_Preference_Score_Better_Aligning_Text-to-Image_Models_with_Human_Preference_ICCV_2023_paper.html.

Xun Wu, Shaohan Huang, Guolong Wang, Jing Xiong, and Furu Wei. Multimodal large language models make text-to-image generative models align better. In *Advances in Neural Information Processing Systems*, volume 37, 2024b. doi: 10.52202/079017-2584. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/9421261e06f1a63a352b068f1ac90609-Abstract-Conference.html.

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. In *Advances in Neural Information Processing Systems*, volume 36, 2023. doi: 10.52202/075280-0700. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/33646ef0ed554145eab65f6250fab0c9-Abstract-Conference.html.

Zeyue Xue, Jie Wu, Yu Gao, Fangyuan Kong, Lingting Zhu, Mengzhao Chen, Zhiheng Liu, Wei Liu, Qiushan Guo, Weilin Huang, and Ping Luo. DanceGRPO: Unleashing GRPO on visual generation. *arXiv preprint arXiv:2505.07818*, 2025. URL https://arxiv.org/abs/2505.07818.

Yankai Yang, Yancheng Long, Hongyang Wei, Wei Chen, Tianke Zhang, Kaiyu Jiang, Haonan Fan, Changyi Liu, Jiankang Chen, Kaiyu Tang, Bin Wen, Fan Yang, Tingting Gao, Han Li, and Shuo Yang. Joint reward modeling: Internalizing chain-of-thought for efficient visual reward models. *arXiv preprint arXiv:2602.07533*, 2026. URL https://arxiv.org/abs/2602.07533.

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. *arXiv preprint arXiv:2505.20275*, 2025. URL https://arxiv.org/abs/2505.20275.

Ya-Qi Yu, Fangyu Hong, Xiangyang Qu, Hao Wang, Gaojie Wu, Qiaoyu Luo, Nuo Xu, Huixin Wang, Wuheng Xu, Yongxin Liao, Zihao Chen, Haonan Li, Ziming Li, Dezhi Peng, Minghui Liao, Jihao Wu, Haoyu Ren, and Dandan Tu. Visual preference optimization with rubric rewards. *arXiv preprint arXiv:2604.13029*, 2026. URL https://arxiv.org/abs/2604.13029.

Shu Zhang, Xinyi Yang, Yihao Feng, Can Qin, Chia-Chih Chen, Ning Yu, Zeyuan Chen, Huan Wang, Silvio Savarese, Stefano Ermon, Caiming Xiong, and Ran Xu. HIVE: Harnessing human feedback for instructional visual editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9026–9036, 2024a. URL https://arxiv.org/abs/2303.09618.

Sixian Zhang, Bohan Wang, Junqiang Wu, Yan Li, Tingting Gao, Di Zhang, and Zhongyuan Wang. Learning multi-dimensional human preference for text-to-image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8018–8027, 2024b. URL https://arxiv.org/abs/2405.14705.

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

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 *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/d8e68ddfe22520b45b8fb8d5cbde5a21-Abstract-Conference.html.

A Implementation Details

A.1 Stage I: SFT Implementation Details

We perform supervised fine-tuning (SFT) using **ms-swift** on 8 NVIDIA A100 GPUs. The training hyperparameters are summarized in Table 4.

Table 4: Hyperparameters for Stage-I supervised fine-tuning.

Hyperparameter	**Value**
Optimizer	AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.999$)
Learning Rate	1×10^{-5}
Learning Rate Scheduler	Linear
Warmup Ratio	0.1
Per-Device Batch Size	1
Gradient Accumulation Steps	4
Effective Global Batch Size	32
DeepSpeed Stage	ZeRO-2
Maximum Sequence Length	32,768
Maximum Image Pixels	802,816

A.2 Stage II: CR-OPSD Implementation Details

We implement CR-OPSD using the **verl** framework. Both the student and teacher are initialized from the Stage-I SFT checkpoint, and the teacher is maintained as an exponential moving average (EMA) of the student with an update rate of 0.01. The training hyperparameters are summarized in Table 5. To accommodate samples with many reference images, we set the maximum image pixel budget to $802{,}816$ pixels per image and reduce it to $262{,}144$ pixels when more than five reference images are provided.

Table 5: Hyperparameters for Stage-II CR-OPSD.

Hyperparameter	**Value**
Global Batch Size	128
Mini-Batch Size	128
Rollouts per Prompt	4
Log-Probability Top-K	10
Maximum Prompt Length	32,768
Maximum Response Length	128
Learning Rate	1×10^{-6}
Learning Rate Scheduler	Cosine
Warmup Ratio	0.03
Training Epochs	1
Training Temperature	1.0
Teacher Temperature	1.0
Teacher EMA Update Rate	0.01
KL Coefficient	0.0
Maximum Image Pixels	802,816

A.3 DiffusionNFT Implementation Details

We use CheckDistill to guide RL post-training of the image editing model with DiffusionNFT (Zheng et al., 2026). For each editing condition h, the old policy generates G candidates. We evaluate each candidate to obtain its reward $R_k = R(x_k)$ using Eq. 8 and 1, which is then used

to determine its positive-branch weight relative to the group mean:

$$r_k = \frac{1}{2} + \frac{1}{2}\,\mathrm{clip}\left(\frac{R_k - \bar{R}_h}{Z_c}, -1, 1\right), \qquad \bar{R}_h = \frac{1}{G}\sum_{k=1}^{G} R_k, \tag{9}$$

where $Z_c > 0$ is the reward normalization scale and the negative-branch weight is $1 - r_k$.

To suppress updates from nearly saturated candidate groups, we follow the Low-STD Group Filtering strategy of UniWorld-V2 (Li et al., 2025).

$$\sigma_h = \sqrt{\frac{1}{G}\sum_{k=1}^{G}(R_k - \bar{R}_h)^2}, \qquad M_h = 1 - \mathbf{1}\left\{\bar{R}_h > \eta_\mu \wedge \sigma_h < \eta_{\mathrm{std}}\right\}. \tag{10}$$

Here, $\bar{R}_h$ and σ_h denote the mean and standard deviation of the rewards in group h, respectively; η_μ and η_{std} are the corresponding filtering thresholds. Groups with high mean reward and low standard deviation are excluded from optimization by setting $M_h = 0$.

Following DiffusionNFT, let v_ω^+ and v_ω^- denote the positive and negative policy branches, and v_k the target velocity of candidate k. The optimization objective is

$$\mathcal{L}_{\mathrm{NFT}}(\omega) = \mathbb{E}\left[M_h \frac{1}{G}\sum_{k=1}^{G}\left(r_k \left\|v_\omega^+ - v_k\right\|_2^2 + (1 - r_k)\left\|v_\omega^- - v_k\right\|_2^2\right)\right]. \tag{11}$$

We enable classifier-free guidance (CFG) during rollout but disable it during policy optimization to reduce training overhead. Training is performed with LoRA, with the hyperparameters summarized in Table 6.

Table 6: Hyperparameters for DiffusionNFT training.

Hyperparameter	**Value**
Guidance Scale	4.0
Training Sampling Steps	16
Training Sampler	DPM2
Evaluation Sampling Steps	40
Noise Level	0.7
Learning Rate	3×10^{-4}
NFT Mixing Coefficient (β)	1×10^{-4}
Per-GPU Batch Size	1
Group Size	8
Number of Groups	48
Group Mean Filtering Threshold	8
Group Std. Filtering Threshold	0.025
LoRA Rank	32
LoRA Alpha	64
Timestep Fraction	0.99
Maximum Gradient Norm	1.0
Optimizer	AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.999$)
Weight Decay	1×10^{-4}
GPUs	$40\times$ NVIDIA A100 (32 for training, 8 for reward model deployment)

A.4 Reward Benchmark Evaluation Details

We evaluate reward prediction on MMRB2, MER-Bench, and EditReward-Bench, none of which overlaps with the data used for CheckDistill. MMRB2 reports single-reference, multi-reference, and overall accuracy, where the overall accuracy is computed as a sample-count-weighted average of the single- and multi-reference subsets rather than their arithmetic mean. MER-Bench reports

ranking accuracy under the 2-pair, 3-pair, and 4-pair settings, with their arithmetic mean used as the overall metric. EditReward-Bench evaluates prompt following (PF), consistency (CS), and overall preference accuracy. For CheckDistill PF is given by the instruction-following score, while CS is computed from the attribute-preservation (AP) and context-preservation (CP) scores using weights of 0.35 and 0.25, respectively, consistent with their weights in the overall reward. The final scalar reward follows Eq. 8 and 1. Except for specialized reward models with their own inference formats, all evaluated models use the same system prompt. Each candidate is scored independently in a pointwise manner, after which pairwise comparisons and multi-candidate rankings are obtained by comparing the resulting scalar scores.

B Implementation Details of GRPO and Token-Level GRPO

GRPO and Token-Level GRPO are initialized from the same SFT checkpoint as CheckDistill. For each input x, they sample G scoring responses $\{y_i\}_{i=1}^{G}$. Let $S_{i,\delta}$ denote the numerical score represented by the sampled score token for dimension δ in response i, and let g_δ be the corresponding ground-truth label, where $\delta \in \mathcal{D} = \mathcal{D}_{\mathrm{SC}} \cup \{\mathrm{PQ}\}$. Following the reward aggregation rule in the main text, the predicted and label-derived aggregate scores are

$$\begin{aligned} R_i &= \left(\sum_{\delta \in \mathcal{D}_{\mathrm{SC}}} w_\delta S_{i,\delta} \right)^{\lambda} S_{i,\mathrm{PQ}}^{1-\lambda}, \\ R_g &= \left(\sum_{\delta \in \mathcal{D}_{\mathrm{SC}}} w_\delta g_\delta \right)^{\lambda} g_{\mathrm{PQ}}^{1-\lambda}. \end{aligned} \tag{12}$$

GRPO: Sequence-Level Advantage. GRPO constructs a response-level reward from the discrepancy between the predicted and label-derived aggregate scores:

$$r_i = 1 - \frac{|R_i - R_g|}{9}, \qquad A_{i,t}^{\mathrm{GRPO}} = r_i - \frac{1}{G} \sum_{j=1}^{G} r_j. \tag{13}$$

The same group-centered advantage is applied to all tokens in response i.

Token-Level GRPO: Dimension-Level Advantage. Token-Level GRPO instead computes a separate reward for each scoring dimension:

$$\begin{aligned} r_{i,\delta} &= 1 - \frac{|S_{i,\delta} - g_\delta|}{9}, \\ A_{i,t}^{\mathrm{Token}} &= r_{i,\delta(t)} - \frac{1}{G} \sum_{j=1}^{G} r_{j,\delta(t)}, \end{aligned} \tag{14}$$

where $\delta(t)$ denotes the scoring dimension associated with score-token position t. The dimension-specific advantage is applied only at the four score-token positions, while non-score positions receive zero policy advantage. Both methods use group-mean centering without standard-deviation normalization.

Training Objective. GRPO and Token-Level GRPO share the same clipped policy objective and differ only in the definition of $A_{i,t}$. Let

$$\rho_{i,t}(\theta) = \frac{p_\theta(y_{i,t} \mid x, y_{i,<t})}{p_{\theta_{\mathrm{old}}}(y_{i,t} \mid x, y_{i,<t})}. \tag{15}$$

The training loss is

$$\begin{aligned} \mathcal{L}_{\mathrm{base}}(\theta; A) = -\mathbb{E}\Bigg[\frac{1}{\sum_i |y_i|} \sum_{i=1}^{G} \sum_{t=1}^{|y_i|} \min \Big(& \rho_{i,t}(\theta) A_{i,t}, \\ & \mathrm{clip}\left(\rho_{i,t}(\theta), 1-\epsilon_l, 1+\epsilon_h\right) A_{i,t} \Big) \Bigg] + \beta \mathcal{L}_{\mathrm{KL}}. \end{aligned} \tag{16}$$

GRPO and Token-Level GRPO use A^{GRPO} and A^{Token}, respectively. Here, $\mathcal{L}_{\mathrm{KL}}$ regularizes the policy against the frozen SFT reference model, ϵ_l and ϵ_h denote the clipping ranges, and β is the KL coefficient.

Training Settings. We train GRPO and Token-level GRPO on CheckDistill-40K using the **verl** framework under the same training configuration. The shared hyperparameters are summarized in Table 7.

Table 7: Hyperparameters for GRPO and Token-level GRPO.

Hyperparameter	**Value**
Training Temperature	1.0
Global Batch Size	128
Mini-Batch Size	128
Rollouts per Prompt	8
Maximum Prompt Length	16,384
Maximum Response Length	128
Learning Rate	1×10^{-6}
Learning Rate Scheduler	Cosine
Warmup Ratio	0.03
KL Coefficient	0.001
PPO Clip Ratio	$\epsilon_l = 0.2,\ \epsilon_h = 0.3$
Maximum Image Pixels	802,816

C Alternative Reverse-KL Formulation

C.1 Preliminaries

On-Policy Self-Distillation (OPSD). OPSD (Zhao et al., 2026) enables a single model to act as both student and teacher by conditioning on different contexts, where the teacher additionally observes privileged information and provides token-level supervision along student-generated trajectories. Given a task input x, the student samples a response $y \sim p_\theta(\cdot \mid x)$. With privileged information z available to the teacher, the student and teacher predictive distributions at the same response prefix $y_{<t}$ are defined as

$$p_S(y_t \mid x, y_{<t}) = p_\theta(y_t \mid x, y_{<t}), \qquad p_T(y_t \mid x, z, y_{<t}) = p_\theta(y_t \mid x, z, y_{<t}). \tag{17}$$

From a policy-gradient perspective, the token-level distillation signal for a sampled token y_t can be expressed as

$$A_t = \mathrm{sg}\left[\log \frac{p_T(y_t \mid x, z, y_{<t})}{p_S(y_t \mid x, y_{<t})}\right], \tag{18}$$

where $\mathrm{sg}[\cdot]$ denotes the stop-gradient operator. A positive A_t reinforces tokens assigned higher likelihood by the teacher than by the student, whereas a negative A_t suppresses them. The corresponding on-policy policy-gradient surrogate objective is

$$\mathcal{L}_{\mathrm{OPSD}}(\theta) = -\mathbb{E}\left[\frac{1}{|y|}\sum_t A_t \log p_S(y_t \mid x, y_{<t})\right]. \tag{19}$$

C.2 Alternative Objective

Our method admits the following alternative reverse-KL objective:

$$\mathcal{L}^{\mathrm{TopK}}_{\mathrm{wRKL}}(\theta) = \mathbb{E}\left[\frac{1}{|y|}\sum_t c_t \sum_{y_t \in \mathcal{T}_t}\left(p_S(y_t \mid x, y_{<t}) \log \frac{p_S(y_t \mid x, y_{<t})}{p_T(y_t \mid x, z, y_{<t})} - p_S(y_t \mid x, y_{<t}) + p_T(y_t \mid x, z, y_{<t})\right)\right]. \tag{20}$$

The term $-p_S + p_T$ corrects the probability-mass difference on the truncated support. When the candidate set covers the entire vocabulary, this term sums to zero, and Eq. 20 reduces to reliability-weighted reverse KL.

C.3 Connection to the Policy-Gradient Objective

We treat the response prefixes, candidate set $\mathcal{T}_t$, teacher distribution, and c_t as fixed when differentiating with respect to θ. For each candidate, the correction term $-p_S$ cancels the extra $+1$ term arising from differentiating $p_S \log(p_S/p_T)$:

$$\begin{aligned}\nabla_\theta \Bigg[p_S(y_t \mid x, y_{<t}) \log \frac{p_S(y_t \mid x, y_{<t})}{p_T(y_t \mid x, z, y_{<t})} - p_S(y_t \mid x, y_{<t}) + p_T(y_t \mid x, z, y_{<t}) \Bigg] \\ = \log \frac{p_S(y_t \mid x, y_{<t})}{p_T(y_t \mid x, z, y_{<t})} \nabla_\theta p_S(y_t \mid x, y_{<t}).\end{aligned} \tag{21}$$

Substituting Eq. 21 into Eq. 20 and using $\nabla_\theta p_S = p_S \nabla_\theta \log p_S$, we obtain

$$\begin{aligned}\nabla_\theta \mathcal{L}_{\text{wRKL}}^{\text{TopK}} &= \mathbb{E}\left[\frac{1}{|y|} \sum_t \sum_{y_t \in \mathcal{T}_t} c_t \log \frac{p_S(y_t \mid x, y_{<t})}{p_T(y_t \mid x, z, y_{<t})} \nabla_\theta p_S(y_t \mid x, y_{<t}) \right] \\ &= -\mathbb{E}\left[\frac{1}{|y|} \sum_t \sum_{y_t \in \mathcal{T}_t} p_S(y_t \mid x, y_{<t})\, c_t \log \frac{p_T(y_t \mid x, z, y_{<t})}{p_S(y_t \mid x, y_{<t})} \nabla_\theta \log p_S(y_t \mid x, y_{<t}) \right].\end{aligned} \tag{22}$$

Defining $\hat{A}_t(y_t) = \text{sg}\Big[c_t p_S(y_t \mid x, y_{<t}) \log \frac{p_T(y_t|x,z,y_{<t})}{p_S(y_t|x,y_{<t})} \Big]$, we obtain the following policy-gradient surrogate with the same gradient:

$$\mathcal{L}_{\text{PG-RKL}}(\theta) = -\mathbb{E}\left[\frac{1}{|y|} \sum_t \sum_{y_t \in \mathcal{T}_t} \hat{A}_t(y_t) \log p_S(y_t \mid x, y_{<t}) \right]. \tag{23}$$

Eq. 23 matches the training objective in the main text and is gradient-equivalent to Eq. 20 at the parameter point where the advantages are computed.

D Checklist Construction and Judge Templates

D.1 Checklist Construction Template

Given one or more reference images and an editing instruction, the checklist constructor generates atomic verification items for Instruction Following (IF), Attribute Preservation (AP), and Context Preservation (CP). Each item contains five key fields: `type`, indicating the evaluation dimension; `object`, specifying the edited or preserved entity; `location`, describing its spatial position; `bbox_2d`, giving the normalized bounding box in the reference image; and `content`, specifying the atomic requirement to be verified. Bounding-box coordinates are normalized to integer values in $[0, 1000]$, and `bbox_2d` is set to `null` for newly added objects when no expected generation region can be determined. For multiple reference images, `ref_image` further identifies the source image containing the object. The checklist is returned as a single JSON list, as specified in the template below.

Checklist Construction Template

```
You are a strict and meticulous expert in designing image-editing checklists. Given one or
    more reference images and an editing instruction, generate checklists that can guide
    subsequent evaluation.

**## Input**
Prompt: {request}
```

```
Image input: one or more reference images <image>

**## Generation Procedure**
1. Inspect the reference image(s): describe key elements that are actually visible in the
     reference image(s), including their location, count, color, shape, pose, and other
     observable attributes. Do not infer or imagine any invisible content.
2. Parse the instruction: decompose the editing request into concrete add/remove/modify
     operations.
3. Generate checklists: decompose each operation into independently verifiable checklist
     items.

**## Checklist Types**
- Instruction Following (IF): the specific editing action that must be performed on an
     object involved in the edit.
- Attribute Preservation (AP): the key attributes of an edited object that should remain
     unchanged during editing.
- Context Preservation (CP): for each object not involved in the edit, set the content to
     Keep this object unchanged.

**## Generation Rules (Must Be Followed)**
1. **Groundedness**: The object, location, and content in every checklist item must be
     directly grounded in either the visible content of the reference image(s) or the
     explicit editing instruction.
Do not introduce objects, attributes, or locations that are unsupported by either the
     reference image(s) or the editing instruction.

2. **Atomicity**: Each checklist item must describe only one independently verifiable
     target. Do not combine multiple operations, objects, or attributes into a single item.

3. **Removal Operations**: Specify only the object to be removed and its location. Do not
     describe how the background should be filled after removal.

4. **BBox Localization**: Each checklist item must include `bbox_2d`, indicating the
     bounding box of the object in the reference image in the format [x1, y1, x2, y2],
     where the coordinates correspond to the top-left and bottom-right corners and are
     normalized to integer values in [0, 1000]. The `bbox_2d` must tightly cover the
     actually visible extent of the object. When multiple reference images are provided,
     additionally use `ref_image` to indicate the 1-based index of the reference image
     containing the object. For an IF item involving a newly added object that does not
     exist in the reference image, set `bbox_2d` to the expected generation region; if the
     expected region cannot be determined, set it to null.

5. **Self-Check**: Before output, verify that every checklist item is directly supported
     by either the reference image(s) or the editing instruction, and that each `bbox_2d`
     is consistent with the actual object location.

**## Output**
Write all checklist items into a single JSON list. The output language must be consistent
     with the editing instruction:
[
  {
    checklist_id: IntType,
    type: IF | AP | CP,
    object: object name,
    location: position in reference image,
    bbox_2d: [x1, y1, x2, y2],
    content: editing instruction or preservation constraint
  },
  ...
]
```

D.2 JUDGE TEMPLATE

We use two complementary judges for semantic consistency (SC) and perceptual quality (PQ). The SC judge performs instance-specific, checklist-guided evaluation using the editing instruction, generated checklist, reference image(s), and edited result. It independently assesses Instruction Following (IF), Attribute Preservation (AP), and Context Preservation (CP), and outputs checklist-level judgments, dimension-wise scores and rationales, and an overall SC score. The PQ judge instead evaluates the edited result alone under a fixed quality rubric covering sharpness, generation artifacts, structural plausibility, lighting and perspective consistency, and boundary blending, and outputs a perceptual-quality score with a concise rationale. The complete prompt templates are provided below.

SC Judge Template

```
You are a strict and consistent expert in image-editing quality evaluation. Your task is
     to assign absolute quality scores for instruction following and context preservation
     of an edited result.

**## Input**
Prompt: {request}
Checklists: {checklists_json}
Image input order: one or more reference images (from the first image to the
     second-to-last image) -> edited result image (the last image)

**## Evaluation Rules (Must Be Followed)**
1. **Checklist Evaluation**: Evaluate every provided checklist item one by one. Do not
     introduce additional evaluation criteria, and do not speculate about details that are
     not visible in the images.
2. **Visible Evidence Only**: Every judgment must be based only on differences that can be
     directly observed between the reference image(s) and the edited result. You may use
     the checklist `bbox_2d` ([x1, y1, x2, y2], normalized to [0, 1000]) to localize the
     corresponding object in the reference image before comparing it with the relevant
     region in the edited result.
3. **Independent Scoring**: Score Instruction Following (IF), Attribute Preservation (AP),
     and Context Preservation (CP) independently. Performance in one dimension must not
     compensate for errors in another.
4. **Stronger Penalty for Incorrect Edits**: Editing the wrong object, introducing
     malformed structures, causing structural collapse, or violating physical plausibility
     should be penalized more heavily than simply failing to perform an intended edit.
5. **Full Satisfaction Criterion**: A checklist item is considered fully satisfied only
     when both conditions hold: the requested edit or preservation constraint is correctly
     satisfied, and the corresponding region is visually free of defects.

**## Scoring Scale (Shared by IF, AP, and CP; Integer Scores from 0 to 9)**
First estimate the fraction of checklist items satisfied within the dimension, then
     determine the score by considering the severity of the observed defects:
- 9: All requirements are fully satisfied, with no visible defects.
- 8: All requirements are satisfied, with only minor imperfections in a small number of
     non-critical aspects.
- 7: All critical requirements are completed, while a few non-critical aspects are not
     fully satisfied.
- 6: Most requirements are satisfied, with noticeable but non-severe defects.
- 5: Roughly two-thirds of the requirements are satisfied, or there is one clear defect.
- 4: Roughly half of the requirements are satisfied, or there are multiple clear defects.
- 3: Only a small portion is satisfied, or there is one severe defect, such as editing the
     wrong object, deformation, or structural collapse.
- 2: Very few requirements are satisfied, or there are multiple severe defects.
- 1: Almost none of the requirements are satisfied, with severe incorrect edits.
- 0: The requested edit or preservation constraint is entirely unfulfilled or completely
     incorrect.

**Discriminability Requirement**: Use the full 0-9 score range. Do not cluster scores
     around 5-7 without sufficient evidence. Scores must clearly distinguish among perfect
     results (9), minor imperfections (7-8), noticeable defects (4-6), and severe failures
     (0-3).

Final `score`: compute an overall score with IF as the primary factor and AP/CP as
     secondary factors. The score may be a floating-point value with one decimal place. If
     IF <= 3, the final score must not exceed 4.

**## Output (Strict JSON Only; Use the Same Language as the Editing Instruction)**
- `ckl reason`: provide a judgment rationale for every checklist item without omission.
- `IF reason`, `AP reason`, `CP reason`, and the overall `reason`: summarize all relevant
     observations within the corresponding dimension, using the object as the grammatical
     subject whenever possible.
- Order observations by their impact on the score, from largest to smallest. Keep the
     explanations concise and non-redundant.

{
  ckl reason: string,
  IF score: <0-9 int>,
  IF reason: string,
  AP score: <0-9 int>,
  AP reason: string,
  CP score: <0-9 int>,
  CP reason: string,
  score: <0-9 float>,
  reason: string
}
```

PQ Judge Template

```
You are a strict and consistent expert in image quality evaluation. Your task is to assign
     an absolute score based solely on the visual quality of the edited result itself,
     independent of whether the editing instruction has been successfully fulfilled.

**## Input**
Prompt: {request}
Image input: edited result image

**## Evaluation Rules (Must Be Followed)**
1. Base all judgments only on directly observable evidence in the image. Do not speculate
     about nonexistent content, and do not over-penalize negligible imperfections.
2. Inspect the image in the following order:
    sharpness / noise / blur
    artifacts / compositing misalignment / unnatural textures
    deformation or structural errors in humans and objects
    consistency of lighting / perspective / color
    edge blending and overall realism
3. First identify the defects that are actually present, then assign a score according to
     the scale below.

**## Scoring Scale (Integer Scores from 0 to 9)**
- 9: Highly realistic and natural, with sharp image quality, no visible defects, and rich
     details.
- 8: Excellent quality, with only one minor imperfection that does not affect the overall
     viewing experience.
- 7: Good quality, with a small number of minor imperfections that do not materially
     affect the overall appearance.
- 6: Acceptable quality, with noticeable imperfections that slightly affect the viewing
     experience.
- 5: Average quality, with multiple imperfections or one clear defect.
- 4: Poor quality, with multiple clear defects, such as local artifacts or mild
     deformation.
- 3: Bad quality, with severe defects such as obvious deformation, structural errors, or
     large-scale artifacts.
- 2: Very poor quality, with multiple severe defects that substantially degrade the visual
     result.
- 1: Severely distorted, with major structural collapse and little practical usability.
- 0: Completely unusable, corrupted, or visually unrecognizable.

**Discriminability Requirement**: Use the full 0-9 score range. Do not cluster scores
     around 5-7 without sufficient evidence. Assign high scores (8-9) when no meaningful
     defects are present, and confidently assign low scores (0-3) when severe defects are
     observed.

**## Output (Strict JSON Only; Use the Same Language as the Editing Instruction)**
The `reason` field should briefly summarize the main defects or strengths that determine
     the score:
{
  score: <0-9 int>,
  reason: string
}
```

E Training Prompt Templates

E.1 Scoring Prompt for SFT and Student

The same scoring prompt is used for Stage-I supervised fine-tuning and for the student model in Stage-II CR-OPSD. Given the editing instruction, one or more reference images, and the edited result, the model independently predicts four absolute scores for Instruction Following (IF), Attribute Preservation (AP), Context Preservation (CP), and Perceptual Quality (PQ). The model outputs the four integer scores in a fixed order, as specified below.

System Prompt

```
You are a strict and consistent image editing quality judge. Based on the editing
     instruction (Prompt), compare the reference image(s) with the result image, and
     assign absolute quality scores along four independent dimensions: Instruction
```

```
     Following (IF), Attribute Preservation (AP), Context Preservation (CP), and
     Perceptual Quality (PQ).

**## Input**
Prompt: {request}
Image input order: one or more reference images (from the first to the second-to-last)
     the result image (the last one)
{image_tags}

**## Judging Rules (must be followed)**
1. **Visible facts only**: All scoring must be based on visual differences directly
     observable between the reference image(s) and the result image. Never speculate about
     details that do not exist in the images, and do not exaggerate minor flaws.
2. **Independent scoring**: IF, AP, CP, and PQ must each be scored independently, with no
     compensation between them.
3. **PQ is Prompt-independent**: PQ evaluates only the result image's own visual quality,
     independent of how well the editing instruction (Prompt) is fulfilled.
4. **Maintain score discrimination**: Make full use of the 0-9 score range and avoid
     clustering scores at 5-7 without justification. 9 indicates perfect performance, 7-8
     indicates minor issues, 4-6 indicates obvious defects, and 0-3 indicates severe
     failure.

**## Evaluation Dimensions**
- **Instruction Following (IF)**: Evaluates whether the editing result fulfills the Prompt
     requirements. Focus on whether the edited object is correct, whether the edited
     content meets the requirements, whether the target attribute modifications are
     accurate, whether the degree of editing is sufficient, and whether there are missed
     edits, wrong edits, or clearly unreasonable results.
- **Attribute Preservation (AP)**: Evaluates whether the attributes of the edited object
     that the Prompt does not require to change remain consistent. Focus on whether
     non-target attributes such as object identity, appearance, color, structure,
     position, pose, quantity, proportion, material, and texture are preserved.
- **Context Preservation (CP)**: Evaluates whether objects not involved in the Prompt, the
     background, and the overall image properties such as color tone, lighting, and
     composition remain consistent with the reference image(s).
- **Perceptual Quality (PQ)**: Evaluates the result image's own visual quality and overall
     impression. Focus on whether the fundamental image quality (sharpness, blurriness,
     noise) is good; whether there are generation artifacts (unnatural textures, stitching
     misalignment, etc.), deformations, or physics-violating phenomena; whether human or
     object structures are plausible; whether lighting, shadows, perspective, reflections,
     and colors are harmonious and consistent; whether the background blends naturally
     with the subject; whether textures and edges look natural; and the overall realism,
     aesthetics, and detail quality.

**## Scoring Scale (integer, 0-9)**

**### Instruction Following (IF)**
9: Fully follows the Prompt. The edited object is correct, all requirements are accurately
     completed, and the result looks natural.
0: The edit is not performed at all, or is performed completely incorrectly: the edited
     object is wrong, the edited content is contrary or irrelevant to the Prompt
     requirements, or the editing result is severely deformed or structurally broken.

**### Attribute Preservation (AP)**
9: The edited object remains highly consistent; only the attributes explicitly required by
     the Prompt are changed, and all non-target attributes are preserved.
0: The non-target attributes of the edited object are severely lost; its identity or
     structure is completely changed and it can no longer be recognized as the same object
     in the reference image(s).

**### Context Preservation (CP)**
9: The non-edited content remains completely consistent, with no observable changes.
0: The non-edited content is largely destroyed or entirely repainted, with almost no
     consistency with the reference image(s).

**### Perceptual Quality (PQ)**
9: Realistic and natural; sharp fundamental quality with no blurring, noise, artifacts,
     deformations, or physics violations; plausible structure; harmonious lighting and
     color; rich detail.
0: Severely distorted; pervasive deformations, generation artifacts, quality defects, and
     physics errors; broken structure; essentially unusable.

**## Output:**
[IF score, AP score, CP score, PQ score]
```

E.2 Privileged Information for the Teacher

In Stage-II CR-OPSD, the teacher receives the same evaluation instruction and image inputs as the student, together with additional instance-specific privileged information $z = (C, J, A)$. Specifically, C denotes the adaptive checklist constructed for the current editing instance, J contains checklist-level judgments grounded in visible evidence, and A provides dimension-wise rationales for IF, AP, CP, and PQ. The additional teacher context is formatted as follows:

Privileged Information

```
## Checklist

{For each checklist item: checklist_id, type, object, location, bbox_2d, content;
include ref_image when multiple reference images are provided.}

## Checklist Judgments

{For each checklist item, provide its judgment and corresponding visible evidence,
aligned by checklist_id.}

## Dimension-wise Rationales

IF: {Rationale for instruction following}
AP: {Rationale for attribute preservation of the edited object}
CP: {Rationale for preservation of non-edited content}
PQ: {Rationale for the perceptual quality of the edited result}
```

F Additional Quantitative Results

F.1 Complete Ablation Results

We provide the complete ablation results on MMRB2, MER-Bench, and EditReward-Bench. Table 8 reports the effect of different privileged-information configurations, while Table 9 presents the full ablation of design choices in CR-OPSD.

Table 8: Privileged-information ablation (%, ↑). Bold marks column maxima.

	MMRB2			MER-Bench				EditReward-Bench		
Privileged Information	**Single**	**Multi**	**Overall**	**2 pair**	**3 pair**	**4 pair**	**Overall**	**PF**	**CS**	**Overall**
Checklist C	72.99	**67.98**	72.10	81.00	56.50	27.00	54.83	**86.60**	84.20	82.10
+ Checklist Judgment (C, J)	73.24	66.29	72.00	81.50	59.50	27.00	56.00	84.60	**85.10**	82.20
+ Dimension-wise Rationale (C, J, A)	**73.36**	67.42	**72.30**	**82.00**	**60.50**	27.50	**56.67**	85.50	84.10	**82.90**
+ Dimension Scores $(C, J, A, \mathbf{g})$	71.17	65.17	70.10	**82.00**	57.50	**28.00**	55.83	84.90	83.90	81.30

Table 9: Ablation study of the design choices in CR-OPSD (%, ↑). All variants use the same privileged information (C, J, A). Bold indicates the best result in each column.

	MMRB2			MER-Bench				EditReward-Bench		
Variant	**Single**	**Multi**	**Overall**	**2 pair**	**3 pair**	**4 pair**	**Overall**	**PF**	**CS**	**Overall**
w/o Reliability Weighting	71.41	63.48	70.00	81.50	57.50	26.00	55.00	84.70	**85.10**	82.10
Linear Reliability Mapping	72.51	65.73	71.30	81.50	59.50	**27.50**	56.17	85.00	84.70	82.30
Argmax-based Reliability	71.90	64.61	70.60	**82.50**	57.50	26.50	55.50	84.60	84.20	82.60
CR-OPSD (Ours)	**73.36**	**67.42**	**72.30**	82.00	**60.50**	**27.50**	**56.67**	**85.50**	84.10	**82.90**

F.2 Complete Image Editing Results

We provide the complete results of the online RL experiments in Tables 10 and 11.

Table 10: Results on ImgEdit-Bench and GEdit-Bench-EN.

	ImgEdit-Bench										GEdit-Bench-EN		
Model	**Add**	**Adjust**	**Extract**	**Replace**	**Remove**	**Background**	**Style**	**Compose**	**Action**	**Overall**	**G_SC**	**G_PQ**	**G_O**
Qwen-Image-Edit	4.40	4.30	3.67	4.64	4.34	4.30	4.83	3.39	**4.71**	4.287	8.855	7.949	8.181
UniWorld-Qwen-Image-Edit	4.42	4.31	3.71	**4.69**	4.32	4.28	4.83	3.42	4.68	4.296	8.824	7.884	8.119
+ DiffusionNFT (CheckDistill-9B)	**4.47**	**4.68**	**4.52**	4.65	**4.41**	**4.40**	**4.89**	**3.60**	4.51	**4.459**	**8.909**	**8.039**	**8.280**

Table 11: Results on RED-Bench.

	RED-Bench														
Model	**Add**	**Adjust**	**Background**	**Beauty**	**Color**	**Compose**	**Extract**	**Low-level**	**Motion**	**Portrait**	**Remove**	**Replace**	**Stylize**	**Text**	**Viewpoint**	**Overall**
Qwen-Image-Edit	4.54	4.08	4.31	3.14	4.06	3.85	3.04	4.47	**4.68**	**4.37**	4.21	**4.59**	4.68	3.94	3.55	4.10
UniWorld-Qwen-Image-Edit	4.48	4.07	4.40	3.19	4.10	3.84	2.87	4.28	4.63	4.35	4.25	4.56	4.72	3.83	3.40	4.06
+ DiffusionNFT (CheckDistill-9B)	**4.60**	**4.25**	**4.42**	**3.37**	**4.38**	**3.99**	**4.07**	**4.59**	4.55	4.10	**4.39**	4.56	**4.73**	**4.15**	**3.73**	**4.26**

G CHECKLIST-GUIDED EVALUATION EXAMPLES

We provide examples of CheckDistill's checklist-guided judgment in Figs. 6, 7, and 8. Given an instance-specific checklist, CheckDistill performs fine-grained verification of each editing requirement and aggregates these item-level judgments into dimension-level assessments and scores.

Create a watercolor painting of this hamster curled up in a cozy, knitted sock.



Checklist-Guided Input:

1. [IF] overall style
   Location: entire image
   Bbox_2d: [0, 0, 1000, 1000]
   Content: Apply a watercolor painting style to the entire image.
2. [IF] knitted sock
   Location: around the hamster
   Content: Add a cozy, knitted sock into the scene.
3. [IF] hamster
   Location: center of the image
   Bbox_2d: [210, 170, 740, 850]
   Content: Change the hamster's pose to be curled up inside the knitted sock.
4. [AP] hamster's fur
   Location: body of the hamster
   Bbox_2d: [210, 170, 740, 850]
   Content: Maintain the golden and light brown fur color as seen in the reference image.
5. [AP] hamster's facial features
   Location: head of the hamster
   Bbox_2d: [370, 350, 600, 600]
   Content: Preserve the hamster's key facial features, specifically the small black eyes and the pink nose.

CheckDistill Output:



Win

Checklist Judgment
1. The entire image is successfully rendered in a watercolor painting style with appropriate texture and color blending.
2. A cozy, knitted sock has been added to the scene, featuring detailed ribbing and texture.
3. The hamster's pose has been changed to be curled up inside the sock, though it is peeking out rather than being fully curled inside.
4. The hamster's fur color is accurately maintained as golden and light brown, consistent with the reference.
5. The hamster's facial features, including the small black eyes and pink nose, are well-preserved.

Dimension Summary
IF (Instruction Following): The watercolor style is perfectly applied. The knitted sock is a high-quality addition. The hamster is positioned inside the sock, fulfilling the core request, although the "curled up" aspect is interpreted as peeking out rather than being fully enclosed.
AP (Attribute Preservation): The hamster's fur color and facial features are identical to the reference image, maintaining excellent consistency.
CP (Context Preservation): The background is completely transformed into a watercolor scene, which is consistent with the style change requested, though the original background details are lost.
PQ (Perceptual Quality): The image exhibits high visual quality with a beautiful watercolor aesthetic. The textures of the knitted sock and the hamster's fur are rendered with excellent detail. The lighting and shadows are harmonious, and there are no visible artifacts or structural deformities.
[9, 9, 9, 9]



Loss

Checklist Judgment
1. overall style: The image successfully adopts a watercolor painting style with soft edges and blended colors.
2. knitted sock: A cozy, knitted object is added, but it is rendered as a thick braided rope or basket rather than a sock.
3. hamster: The hamster is curled up inside the knitted object, fulfilling the pose requirement.
4. hamster's fur: The golden and light brown fur color is well-preserved.
5. hamster's facial features: The small black eyes and pink nose are accurately maintained.

Dimension Summary
IF (Instruction Following): The watercolor style is applied perfectly. The hamster's pose is correct. However, the knitted object is clearly a rope or basket, failing to represent a "sock" as requested.
AP (Attribute Preservation): The hamster's fur color and facial features (eyes and nose) are consistent with the reference image.
CP (Context Preservation): The background is completely changed to a watercolor wash, which is consistent with the style change but represents a total departure from the original scene.
PQ (Perceptual Quality): The image has high visual quality with excellent watercolor textures, harmonious lighting, and a pleasing composition. The hamster's anatomy is well-proportioned, and the knitted texture is detailed and realistic within the artistic style.
[6, 9, 9, 9]

Figure 6: Example of Checklist-Guided Judgment by CheckDistill

Create a humorous advertisement poster with all three objects together: place the sunglasses from the first image on the Minion figure from the third image, and have the Minion sitting inside the decorative bowl from the second image. Set the scene with a bright, playful background and add bold text that says, "When style, comfort, and fun collide: Summer essentials for everyone!"







Checklist-Guided Input:

1. [IF] sunglasses
   Location: on the Minion figure
   Bbox_2d: [31, 304, 998, 714]
   Reference: 1
   Content: Place the sunglasses from the first image on the Minion figure from the third image.
2. [IF] Minion figure
   Location: inside the decorative bowl
   Bbox_2d: [287, 188, 734, 715]
   Reference: 3
   Content: Place the Minion figure sitting inside the decorative bowl from the second image.
3. [IF] background
   Location: entire scene
   Content: Set the scene with a bright, playful background.
4. [IF] text
   Location: on the poster
   Bbox_2d: [70, 880, 110, 920]
   Content: Add bold text that says, "When style, comfort, and fun collide: Summer essentials for everyone!"
5. [AP] sunglasses
   Location: from the first image
   Bbox_2d: [31, 304, 998, 714]
   Reference: 1
   Content: Maintain the round shape, gold metal frame, and iridescent blue-green lenses of the sunglasses.
6. [AP] decorative bowl
   Location: from the second image
   Bbox_2d: [0, 181, 951, 808]
   Reference: 2
   Content: Maintain the white ceramic material, the specific green leaf/vine pattern, and the three small legs of the bowl.
7. [AP] Minion figure
   Location: from the third image
   Bbox_2d: [287, 188, 734, 715]
   Reference: 3
   Content: Maintain the Minion's yellow skin, single eye, and the black and white striped prisoner outfit with the '001' badge.

CheckDistill Output:



Win

Checklist Judgment
1. The sunglasses are placed in the bowl rather than on the Minion figure as requested.
2. The Minion figure is correctly placed inside the decorative bowl.
3. The background is bright and playful with blue and yellow colors.
4. The text is present but is not bold as specified in the instruction.
5. The sunglasses maintain the round shape and gold frame, but the iridescent blue-green lenses are now a solid dark blue.
6. The decorative bowl maintains its white ceramic material, green leaf pattern, and three legs.
7. The Minion figure maintains its yellow skin, single eye, and prisoner outfit with the '001' badge.

Dimension Summary
IF (Instruction Following): The Minion is correctly placed in the bowl, and the background and text are present. However, the sunglasses were placed in the bowl instead of on the Minion, and the text lacks the requested bold formatting.
AP (Attribute Preservation): The decorative bowl and Minion figure are highly consistent with their reference images. The sunglasses are consistent in shape and frame but lost their specific iridescent lens quality.
CP (Context Preservation): There are no non-edited regions to evaluate for consistency as the entire scene was generated based on the prompt.
PQ (Perceptual Quality): The image is clear and has vibrant colors. However, the sunglasses appear to float without visible support, and the contact between the Minion and the bowl looks physically implausible, reducing the overall visual realism.
[6, 8, 9, 5]



Loss

Checklist Judgment
1. The sunglasses are not placed on the Minion figure; they are resting on the rim of the bowl instead.
2. The Minion figure is not sitting inside the bowl; it is standing on the surface next to the bowl.
3. The background is bright and playful, successfully following the instruction.
4. The text is incomplete and misspelled; it reads "summer essentials [garbled] everyone!" instead of the full required phrase.
5. The sunglasses maintain their round shape, gold frame, and iridescent lenses, though they are in the wrong location.
6. The decorative bowl maintains its white ceramic material, green leaf pattern, and three legs, though the Minion is not inside it.
7. The Minion figure maintains its yellow skin and prisoner outfit, but it has two eyes instead of the single eye from the reference, and it is not inside the bowl.

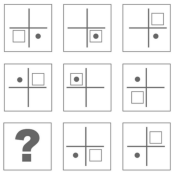
Dimension Summary
IF (Instruction Following): The core spatial instructions were failed: the sunglasses are on the bowl, not the Minion, and the Minion is outside the bowl, not inside it. Additionally, the required text was truncated and contains a garbled artifact.
AP (Attribute Preservation): The sunglasses and the decorative bowl are visually consistent with the reference images. However, the Minion figure lost a key attribute (single eye) and was placed in the wrong location.
CP (Context Preservation): The background and overall composition are consistent with the playful theme requested.
PQ (Perceptual Quality): The image has significant quality issues: the text at the bottom is corrupted and unreadable, the Minion has a distorted face with two eyes, and the sunglasses are poorly integrated into the scene with unnatural lighting and reflections.
[4, 6, 9, 4]

Figure 7: Example of Checklist-Guided Judgment by CheckDistill

H Additional Qualitative Evaluation

We provide additional qualitative results to further illustrate the effectiveness of CheckDistill-guided RL. Figure 9 presents the training reward curve, which steadily increases during optimization. Figure 10 and 11 provide qualitative comparisons before and after CheckDistill-guided RL, showing improved instruction following, stronger source-image consistency, and better perceptual quality.

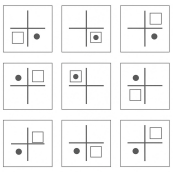
Draw the shape represented by the question mark that fits the pattern.



Checklist-Guided Input:

1. [IF] question mark
   Location: bottom-left cell of the 3x3 grid
   Bbox_2d: [78, 740, 245, 935]
   Content: Remove the question mark from the bottom-left cell.
2. [IF] cross lines
   Location: bottom-left cell of the 3x3 grid
   Bbox_2d: [18, 708, 315, 985]
   Content: Add a central vertical line and a central horizontal line to divide the bottom-left cell into four equal quadrants, matching the structure of the other cells.
3. [IF] square
   Location: bottom-left quadrant of the bottom-left cell
   Bbox_2d: [40, 850, 140, 950]
   Content: Draw a hollow square in the bottom-left quadrant of the cell to follow the counter-clockwise movement pattern observed in the third row (Bottom-Left -> Bottom-Right -> Top-Right).
4. [IF] dot
   Location: bottom-left quadrant of the bottom-left cell
   Bbox_2d: [70, 880, 110, 920]
   Content: Draw a solid dot in the bottom-left quadrant of the cell (inside the square) to follow the static position pattern observed in the third row (Bottom-Left in all three cells).
5. [AP] square
   Location: bottom-left cell
   Bbox_2d: [40, 850, 140, 950]
   Content: The newly drawn square must match the size, line thickness, and gray color of the squares in the other cells, such as the one at [850, 50, 950, 150].
6. [AP] dot
   Location: bottom-left cell
   Bbox_2d: [70, 880, 110, 920]
   Content: The newly drawn dot must match the diameter and solid gray color of the dots in the other cells, such as the one at [200, 190, 225, 215].
7. [AP] cross lines
   Location: bottom-left cell
   Bbox_2d: [18, 708, 315, 985]
   Content: The new cross lines must match the stroke width and color of the grid lines in the existing cells.
8. [CP] other 8 cells
   Location: top two rows and right two columns of the grid
   Bbox_2d: [18, 18, 985, 670]
   Content: Keep the content and layout of the other eight cells in the grid unchanged.

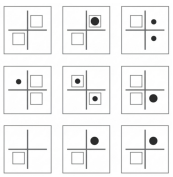
CheckDistill Output:



Win

Checklist Judgment
1. The question mark in the bottom-left cell was successfully removed.
2. The central vertical and horizontal lines were added to the bottom-left cell, matching the grid structure.
3. A hollow square was drawn in the top-right quadrant, but the pattern logic is flawed: the third row should follow a sequence of Bottom-Left -> Bottom-Right -> Top-Right, so the missing first cell should be Bottom-Left, not Top-Right.
4. A solid dot was drawn in the bottom-left quadrant, but it is placed outside the square, whereas the required configuration places the dot inside the square.
5. The square's size, line thickness, and color are consistent with other cells.
6. The dot's diameter and color are consistent, but the square's position relative to the dot is incorrect.
7. The cross lines match the stroke width and color of the existing grid.
8. The other eight cells in the grid are perfectly preserved.

Dimension Summary
IF (Instruction Following): The model failed to correctly deduce the logical pattern for the missing cell. It placed the square in the top-right quadrant, which contradicts the established counter-clockwise movement pattern of the third row (which should result in a Bottom-Left position). Additionally, the dot was placed outside the square, violating the required square–dot configuration.
AP (Attribute Preservation): The visual attributes of the newly added square, dot, and cross lines are perfectly consistent with the style and quality of the other elements in the grid.
CP (Context Preservation): All other cells in the grid were kept exactly as they were in the reference image, with no unintended modifications.
PQ (Perceptual Quality): The image is clean, sharp, and free of artifacts. The lines are straight, and the shapes are geometrically perfect.
[5, 9, 9, 9]



Loss

Checklist Judgment
1. The question mark was removed, but the cell was not filled with the correct pattern; instead, the entire grid was corrupted.
2. The cross lines were added to the bottom-left cell, but the other cells were also modified, violating the instruction to only edit the target cell.
3. A hollow square was drawn in the bottom-left quadrant, but the other cells in the grid were changed, and the pattern logic was not followed correctly across the whole image.
4. A solid dot was not drawn in the bottom-left quadrant, and the other cells were altered.
5. The square in the bottom-left cell matches the style of the other squares, but the overall grid consistency is lost.
6. The dot in the bottom-left cell is missing, so its style cannot be assessed, but the overall grid consistency is lost.
7. The cross lines in the bottom-left cell match the style of the other lines, but the other cells were also modified.
8. The other eight cells were not kept unchanged; the entire grid was regenerated with incorrect content.

Dimension Summary
IF (Instruction Following): The model failed to follow the core instruction of solving the puzzle for the specific cell. Instead of drawing the correct shape to complete the pattern in the bottom-left cell, it completely regenerated the entire grid with incorrect and nonsensical content.
AP (Attribute Preservation): The visual attributes of the newly drawn square and cross lines are consistent with the style of the other elements, but since the entire grid was changed, the context is lost.
CP (Context Preservation): The other eight cells were not preserved; they were all modified and contain incorrect content compared to the reference image.
PQ (Perceptual Quality): The image is clean and sharp with no visible artifacts or blurring. The lines and shapes are well-defined, and the overall visual quality is high.
[1, 9, 1, 9]

Figure 8: Example of Checklist-Guided Judgment by CheckDistill

I DISCUSSION

Z-Reward (Jin et al., 2026) and our CheckDistill both distill reasoning-supported assessment into direct scoring, but address different challenges. Z-Reward focuses on representing preference uncertainty and fine-grained score differences through reasoning-conditioned score distributions. **Our starting point is that image-editing evaluation requires instance-specific verification of both requested changes and preservation constraints.** We therefore introduce Dynamic Checklist Scaling

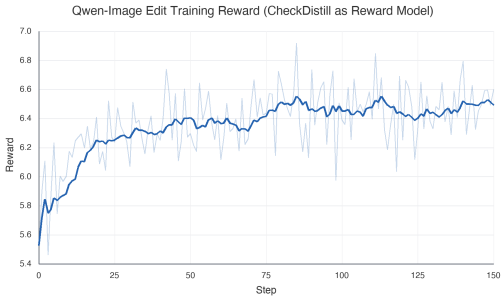


Figure 9: Training reward for Qwen-Image-Edit using CheckDistill as the reward model.

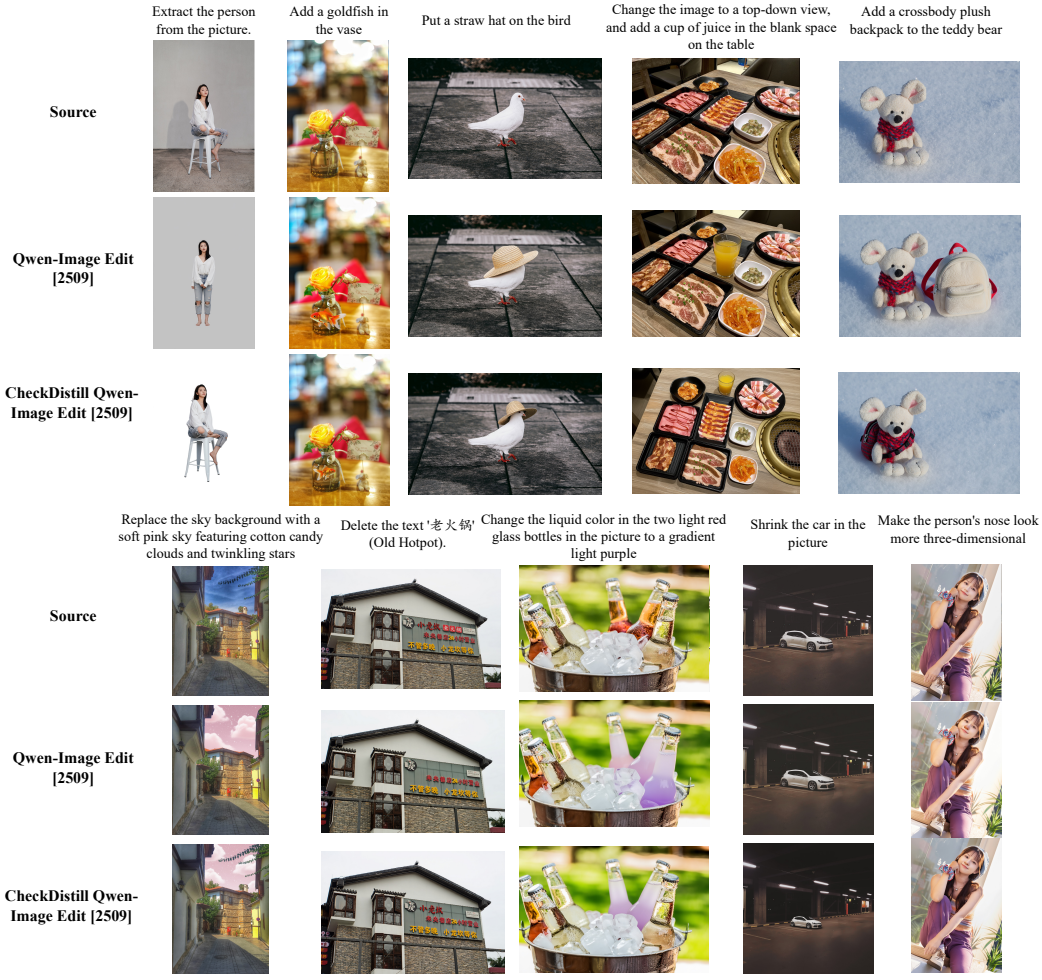


Figure 10: Qualitative Comparison Before and After CheckDistill-based Reward Training.

to translate each editing instruction and source image into atomic verification criteria, enriching the teacher context with checklists and their supporting judgements.

This focus on verification evidence also shapes our distillation design. While Z-Reward transfers scoring knowledge from a large reasoning teacher to a compact student, we use on-policy self-distillation with an evidence-enriched teacher of the same architecture. Distribution-Aligned Context Compression transfers the teacher's scoring behavior at student-generated prefixes, and relia-



Figure 11: Qualitative Comparison Before and After CheckDistill-based Reward Training.

bility weighting modulates supervision at score positions according to teacher–label agreement, accounting for potentially incorrect teacher judgements. **Our contribution centers on constructing and reliably internalizing instance-specific verification evidence**, enabling direct scoring without checklist inputs or explicit reasoning at inference time.