APT: FEEDBACK-DRIVEN AGENTIC POST-TRAINING FOR TEXT-TO-IMAGE GENERATION

Anonymous authors
Paper under double-blind review

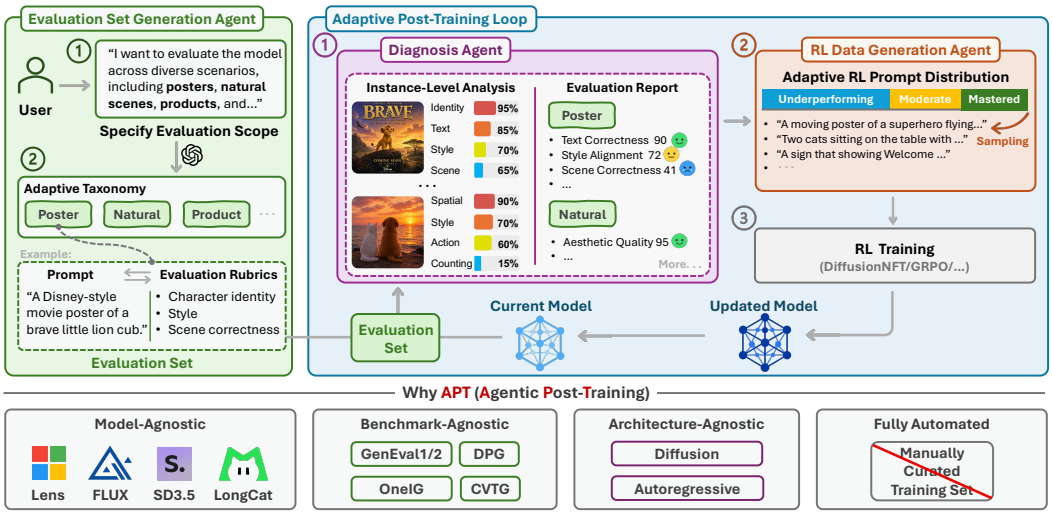


Figure 1: **APT (Agentic Post-Training)** turns post-training into a fully automated, evaluation-driven adaptive loop. Given a user-specified evaluation scope, the **Evaluation Set Generation Agent** constructs a diverse evaluation set consisting of *prompt–evaluation rubric pairs*; the **Diagnosis Agent** evaluates the current model on these pairs to identify its strengths and failure modes; and the **RL Data Generation Agent** adaptively samples next-round training prompts, prioritizing underperforming capabilities. The updated model is then re-evaluated in the next round, allowing the training distribution to continuously adapt to the model's evolving capability profile. This design eliminates the need for manually curated post-training data and makes APT *model-agnostic*, *benchmark-agnostic*, *architecture-agnostic*, and *fully automated*.

ABSTRACT

Post-training is crucial for improving text-to-image models, yet existing pipelines typically rely on manually curated training data and fixed training distributions. We introduce APT (Agentic Post-Training), a fully automated framework that adaptively post-trains a base model according to its evolving capability profile. APT coordinates three agents to construct evaluation sets from user requirements, diagnose model-specific strengths and failure modes, and generate reinforcement learning data with an adaptive prompt distribution that emphasizes underperforming capabilities. By repeatedly evaluating, diagnosing, and updating the generator, APT lets the current model's behavior shape its next training tasks, forming a feedback loop inspired by recursive self-improvement. For general-purpose settings without user-specified evaluation scenarios, we further develop an evaluation harness that enables broad and diverse evaluation set generation. Experiments across multiple base models, benchmarks, and generation architectures demonstrate that APT is model-agnostic, benchmark-agnostic, architecture-agnostic, and fully automated.

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.

1 INTRODUCTION

Recent text-to-image (T2I) models have achieved remarkable progress in visual quality, prompt following, text rendering, compositional generation, and controllability (Black Forest Labs, 2025; OpenAI, 2026; Google, 2026; SeedDance, 2026; Zhao et al., 2026). Beyond large-scale pre-training, post-training has become increasingly important for correcting specific failure modes and aligning model behavior with desired objectives (Xu et al., 2026; Chen et al., 2026a; Jiang et al., 2026a; Feng et al., 2026b). Existing post-training pipelines, however, typically rely on manually constructed training distributions: practitioners identify model weaknesses, collect or synthesize corresponding training data, and repeatedly adjust the data distribution based on empirical observations (Xu et al., 2026; Chen et al., 2026a; Feng et al., 2026b; Geng et al., 2025). Such a workflow is labor-intensive and difficult to scale as models, benchmarks, and user requirements continue to diversify.

More fundamentally, we argue that post-training should be *model-adaptive*. Different base models exhibit substantially different capability distributions: a model that performs well on text rendering may struggle with counting or spatial relations, while another may exhibit the opposite behavior. Consequently, a post-training dataset that is effective for one model is not necessarily optimal for another. We can view each base model M as inducing its own capability and failure distribution p_M over generation tasks. Effective post-training should therefore identify where the current model underperforms, sample training examples accordingly (Chen et al., 2026b), and continuously adapt the sampling distribution as the model improves (Li et al., 2026). In other words, the key question is not only *how to optimize a model*, but also *what the current model should be trained on*. This perspective naturally connects post-training to recursive self-improvement (RSI), in which the evaluation results of the updated model are used to determine the training tasks for the next round (Zelikman et al., 2023; Zweiger et al., 2026; Huang et al., 2026).

To address this problem, we introduce **APT (Agentic Post-Training)**, a fully automated framework that turns post-training into an adaptive, evaluation-driven self-improvement loop, as illustrated in Figure 1. Given a T2I base model and an evaluation requirement, APT automatically constructs an evaluation set of prompt–rubric pairs, diagnoses the current model's strengths and weaknesses, and generates model-specific RL training data. The updated model is then re-evaluated, allowing its evolving behavior to continuously reshape the training distribution for the next round. Rather than relying on a fixed, manually curated dataset, APT lets the current model determine what should be learned next.

As shown in Figure 1, APT is built around three collaborative agents. The *Evaluation Set Generation Agent* converts a high-level evaluation scope into a structured evaluation suite with an adaptive taxonomy, prompts, and task-specific rubrics. The *Diagnosis Agent* evaluates the current model at the instance and category levels and summarizes systematic strengths and failure modes. The *RL Data Generation Agent* then converts this diagnosis into an adaptive prompt distribution that prioritizes underperforming capabilities while retaining moderate and mastered ones, producing model-specific data for RL-based post-training (Zheng et al., 2026; Liu et al., 2026a; Shao et al., 2024). Repeated evaluation, diagnosis, and adaptive sampling provide an implicit mechanism for discovering and tracking the evolving model-dependent distribution p_M without explicitly estimating it.

APT also supports general-purpose post-training when no evaluation scope is specified. For this setting, we develop an **evaluation harness** that guides the Evaluation Set Generation Agent to systematically construct a broad and diverse evaluation set spanning common generation scenarios and capability dimensions (Feng et al., 2026c).

The resulting framework is *model-agnostic*, *benchmark-agnostic*, *architecture-agnostic*, and *fully automated*, and we validate these properties across multiple T2I models, generation architectures, and benchmarks (Chen et al., 2026a; Esser et al., 2024; Black Forest Labs, 2024; Team et al., 2025; Chen et al., 2025). APT consistently improves diverse base models and outperforms post-training with a carefully handcrafted RL training set. Moreover, transferring APT-generated training data across models is consistently inferior to adaptively generating data for the target model, supporting our central hypothesis that effective post-training distributions are model-specific and should evolve together with the model. Together, these results establish agentic, evaluation-driven adaptation as a practical step toward recursive self-improvement for text-to-image generation.

2 METHOD

Overview. Figure 1 provides an overview of APT. The Evaluation Set Generation Agent is invoked once before post-training to construct the fixed evaluation set $\mathcal{E}$ of prompt–rubric pairs (Section 2.1). Users may specify an evaluation scope; when no scope is provided, we use our general-purpose evaluation harness to ensure broad and comprehensive coverage. APT then enters an iterative post-training loop. At each round, the Diagnosis Agent (Section 2.2) evaluates the current model and produces an evaluation report, which is passed to the RL Data Generation Agent (Section 2.3) to generate training data for the next RL update. This RSI-inspired loop repeats after each update, enabling the training distribution to adapt to the model's evolving weaknesses. As described in Section 2.4, APT is model-agnostic, benchmark-agnostic, and architecture-agnostic, and can therefore be applied to diverse text-to-image models and evaluation settings.

Round-Wise Learning Objective. At the beginning of round t, M_t denotes the current text-to-image generator; p and x denote a text prompt and a generated image, respectively. Let D_t be the generated dataset of training prompts for this round, and write $(p, R) \sim D_t$ to denote sampling a prompt–rubric-set pair from this dataset. For a prompt p_i, let R_i be its associated rubric set and $\rho \in R_i$ an individual evaluation criterion. Both diagnosis and training use GPT-4.1-mini as vision-language evaluator, denoted by V. Its binary output $V(x, p, \rho) \in \{0, 1\}$ is 1 if image x satisfies criterion ρ for prompt p, and 0 otherwise. For an image x_i generated from p_i, the training reward is the arithmetic mean of these rubric-level scores:

$$r(x_i, p_i) = \frac{1}{|R_i|} \sum_{\rho \in R_i} V(x_i, p_i, \rho).$$

Each prompt's rubric set is fixed and shared across all its generated images within a rollout round. The evaluator's scoring instructions are provided in Appendix E.3. The round-wise expected-reward objective is

$$J(M_t) = \mathbb{E}_{(p,R)\sim D_t,\, x\sim M_t(\cdot|p)}[r(x, p)]\,. \tag{1}$$

Starting from M_t, the configured RL algorithm takes a finite number of optimization steps aimed at increasing $J(M_t)$, with D_t and the reward procedure fixed within the round.

Recursive Data–Model Coupling. APT couples an *outer loop of diagnostic-driven task adaptation* with an *inner loop of RL-based model optimization*. At round t, $\mathcal{B}_t \subset \mathcal{E}$ is a sampled diagnostic batch of prompt–rubric pairs, and $\mathcal{R}_t$ denotes the resulting rubric-level evaluation report for M_t. Let Diag and $\mathcal{G}$ denote the diagnosis and data-generation procedures, respectively, and let $\mathcal{U}$ denote the RL update operator. The coupled update is

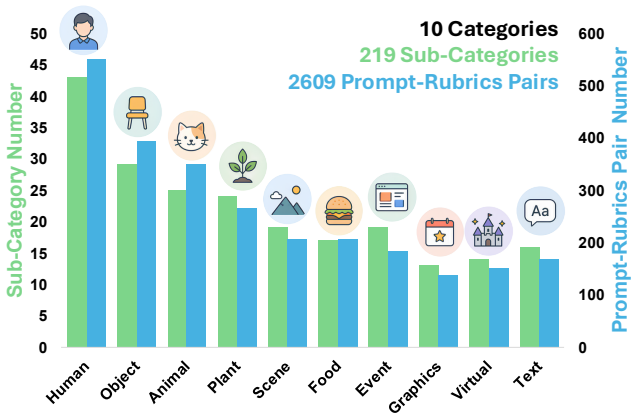
$$\begin{aligned} \mathcal{R}_t &= \mathrm{Diag}(M_t, \mathcal{B}_t), \qquad D_t = \mathcal{G}(\mathcal{R}_t), \\ M_{t+1} &= \mathcal{U}(M_t, D_t; V). \end{aligned} \tag{2}$$

The update operator $\mathcal{U}$ returns M_{t+1} by applying the configured finite number of RL steps to M_t, with the aim of maximizing $J(M_t)$ in Equation 1. Re-evaluating M_{t+1} yields a diagnostic report $\mathcal{R}_{t+1}$ that guides the generation of training prompts D_{t+1} for the next RL update. We next describe the three agents that instantiate this feedback-driven post-training loop.

2.1 EVALUATION SET GENERATION AGENT

The Evaluation Set Generation Agent is responsible for constructing the evaluation data used to diagnose the current model before the post-training stage. When a user specifies an evaluation scope, the agent directly generates evaluation examples tailored to the requested scenarios. For the more general setting where no evaluation scope is provided, we further design a *general-purpose evaluation harness* that guides the agent to systematically construct a broad-coverage evaluation set spanning diverse image-generation scenarios and capabilities. Each evaluation example consists of an image-generation prompt paired with its corresponding prompt-specific evaluation rubrics. The evaluation set is constructed through a multi-step generation pipeline consisting of scenario generation, prompt generation, and rubric generation.

Step 1: Scenario Generation. We begin with a broad set of common image-generation scenario categories, including Human, Object, Animal, Plant, Scene, Food, Event, Fictional World, Text,



Prompt-Rubrics Pair Example

Prompt:

"Two cream-colored puppies sit side by side on a brushed steel exam table..."

Rubrics:

- *"Verify the puppies have a cream-colored coat."*
- *"Verify that exactly two puppies are present as specified in the prompt."*
- *"Verify the two puppies are sitting side by side on the brushed steel exam table."*
- ...

Figure 2: *Left*: Statistics of the sub-categories within each category and the prompt-rubrics pairs within each category. All prompt-rubrics pairs are automatically generated by our Evaluation Set Generation Agent, providing comprehensive coverage for evaluating the current T2I model. *Right*: An example prompt-rubrics pair.

and UI and Graphic Design. For each category, we prompt an LLM to generate dozens of fine-grained sub-categories and corresponding items. For example, the *Human* category may include sub-categories such as *Race*, *Occupation*, and *Gender*, while the *Occupation* sub-category may further contain items such as *Researcher* and *Doctor*. This process produces a hierarchical scenario taxonomy that provides broad and structured coverage of diverse image-generation scenarios.

Step 2: Prompt Generation. Given the items derived from the scenario taxonomy, we further enrich them with a set of descriptive dimensions to generate diverse image-generation prompts. These descriptions capture common compositional and semantic requirements, such as attributes, spatial relationships, object counts, interactions, and colors. For each item, we randomly sample one to four descriptions and prompt GPT-4.1, together with a predefined system prompt described in Appendix E.1, to synthesize the final image-generation prompt. For example, an item such as *Doctor* can be combined with descriptions of color, interaction, and spatial relationship to produce a more compositional and challenging prompt. This design increases both the diversity and coverage of the generated evaluation prompts.

Step 3: Rubric Generation. For each generated prompt, we further construct prompt-specific evaluation rubrics (Feng et al., 2026c). Concretely, we feed the prompt, together with the rubric-generation system prompt described in Appendix E.2, into GPT-4.1-mini to generate up to 10 prompt-specific rubrics. *These rubrics specify the aspects that should be checked for the given prompt*, such as identity consistency, text correctness, style alignment, scene correctness, spatial relations, or counting accuracy, depending on the prompt content. As a result, each evaluation example is represented as a prompt paired with its corresponding rubrics, enabling fine-grained and task-aware diagnosis of model behavior.

Overall, the Evaluation Set Generation Agent transforms a high-level evaluation requirement into a diverse evaluation set containing 2,609 prompt–rubrics pairs. Detailed statistics are provided in Figure 2. This evaluation set serves as the foundation for the subsequent Diagnosis Agent, which analyzes model performance and drives the adaptive post-training loop.

2.2 Diagnosis Agent

The Diagnosis Agent analyzes the performance of the current model on the evaluation set and summarizes its strengths and weaknesses into a structured evaluation report. At each post-training round t, we randomly sample a batch $\mathcal{B}_t = \{(p_i, R_i)\}_{i=1}^{N} \subset \mathcal{E}$ of prompt–rubric pairs, where N is the diagnostic batch size (120 by default). The diagnosis process consists of two stages: *instance-level analysis* and *evaluation report construction*.

Instance-Level Analysis. Within round t, let $K_i = |R_i|$ be the number of rubrics for the i-th sampled prompt p_i, and let ρ_{ij} denote its j-th rubric. We generate an image $x_i^{(t)}$ with the current

model M_t and use the same evaluator V as in the training reward to obtain the rubric score $s_{ij}^{(t)}$:

$$x_i^{(t)} \sim M_t(\cdot \mid p_i), \qquad s_{ij}^{(t)} = V(x_i^{(t)}, p_i, \rho_{ij}) \in \{0, 1\}. \tag{3}$$

This yields a set of prompt-specific rubric-level evaluation results $E_i^{(t)} = \{(\rho_{ij}, s_{ij}^{(t)})\}_{j=1}^{K_i}$, providing a fine-grained diagnosis of the current model for prompt p_i.

Evaluation Report Construction. After processing all N sampled prompt–rubric pairs, the Diagnosis Agent constructs the report $\mathcal{R}_t = \{(p_i, E_i^{(t)})\}_{i=1}^{N}$, instantiating the diagnosis procedure Diag in Equation 2. Each entry in $\mathcal{R}_t$ contains the original prompt together with its rubric-level evaluation results, where each result consists of an evaluation rubric and its corresponding binary score. The resulting report captures the current model's successes and failures across diverse generation capabilities and is subsequently passed to the RL Data Generation Agent to guide the construction of training data for the subsequent RL update.

2.3 RL Data Generation Agent

The RL Data Generation Agent takes the evaluation report produced by the Diagnosis Agent as input and generates training prompts for the subsequent RL update. Its main role is to identify recurring failure patterns in the report, summarize the capabilities that should be improved, and translate these findings into a model-specific training distribution.

We use the evaluation report as an empirical characterization of the current model's capability distribution. This is feasible for several reasons. First, the evaluation set is designed to provide broad coverage over diverse generation scenarios and capabilities. As a result, the sampled evaluation examples probe many different regions of the model's capability space rather than concentrating on a narrow benchmark-specific domain. Second, each prompt is associated with multiple prompt-specific rubrics, which provide fine-grained feedback on *what* the model succeeds or fails on. Therefore, the report contains not only whether a generation is successful, but also structured evidence about the underlying capability responsible for the success or failure. Third, modern language models such as GPT-5.4 have strong summarization and reasoning capabilities, allowing them to compare observations across many evaluation examples and identify recurring patterns that may not be obvious from any individual sample. For instance, prompts from different semantic categories may repeatedly fail for the same underlying reason, such as accurate counting, rendering text on objects, or preserving specific attributes. By aggregating these cross-example patterns, the agent can infer which capabilities are consistently underperforming and which are already handled reliably. Finally, because the evaluation report is regenerated from the current model at every post-training round, the inferred capability profile naturally changes as the model improves. In this sense, the Diagnosis Agent and RL Data Generation Agent together provide an implicit and continually updated estimate of the model's current performance distribution, without requiring an additional explicitly trained estimator.

Based on the summarized failure patterns, the RL Data Generation Agent generates N_{train} new prompts for the current round's RL update, with $p_{t,k}^{\text{train}}$ denoting the k-th generated prompt and $R_{t,k}^{\text{train}}$ denoting its corresponding rubric set generated by GPT-4.1-mini using the same procedure and system prompt as for the evaluation set :

$$D_t = \mathcal{G}(\mathcal{R}_t) = \{(p_{t,k}^{\text{train}}, R_{t,k}^{\text{train}})\}_{k=1}^{N_{\text{train}}}. \tag{4}$$

This instantiates the data-generation procedure $\mathcal{G}$ in Equation 2, conditioned on the current model's diagnostic report $\mathcal{R}_t$. A clear majority of these prompts target recurring weak capabilities, while a smaller portion covers moderately performing and already-strong capabilities to preserve diversity and avoid over-concentrating on a single failure mode. Since different base models exhibit different patterns in the same broad evaluation space, this process naturally produces model-specific training data. Moreover, as the model evolves over successive rounds, the generated training distribution is updated accordingly.

We implement the RL Data Generation Agent with GPT-5.4. The detailed instruction prompt used by the agent is provided in Appendix E.4.

2.4 General, Adaptive, and Fully Automated Post-Training

APT is designed as a general post-training framework that can be applied across different evaluation settings, base models, and generation architectures without task-specific or model-specific redesign. Specifically, APT is (1) *benchmark-agnostic*: it does not require independently designed prompt sets for individual benchmarks, and a single APT run can be evaluated across multiple T2I benchmarks; (2) *model-agnostic*: the same agentic workflow can be directly applied to diverse base models, including Lens (Chen et al., 2026a), SD3.5 (Esser et al., 2024), FLUX (Black Forest Labs, 2024), and LongCat-Image (Team et al., 2025); (3) *architecture-agnostic*: APT is not tied to a particular image-generation formulation and is effective for both diffusion-based and autoregressive image-generation models (Chen et al., 2025; Sun et al., 2024); and (4) *fully automated*: the evaluation results of the current model directly drive the generation of prompts for subsequent RL updates, eliminating the need for manually inspecting failures, designing training distributions, or curating model-specific post-training data (Xu et al., 2026; Geng et al., 2025).

Recursive Self-Improvement for Text-to-Image Generation. APT brings recursive self-improvement (RSI) (Zelikman et al., 2023; Zweiger et al., 2026) to text-to-image generation by making the generator's evolving capabilities drive the construction of its own training curriculum. As formalized in Equation 2, the current model produces behavioral evidence that agents translate into targeted training tasks; RL updates the generator on these tasks, and the updated generator provides fresh evidence for the next round. This creates a recursive coupling between model optimization and task construction: learning changes the model's capability profile, which in turn changes what the system trains on next. Through this feedback loop, APT continually revises its training priorities in response to the generator's remaining and newly exposed weaknesses. By automating diagnosis, curriculum construction, and RL updates, APT turns the generator's own successes and failures into actionable signals for directing subsequent improvement. APT thus operationalizes RSI for T2I post-training through an adaptive curriculum that evolves alongside the model, replacing manual intervention in deciding what the model should learn next.

3 Experiments

3.1 Experimental Setup

Base T2I Models. To demonstrate that APT is model-agnostic, we apply it to four diffusion-based T2I models: *Lens* (Chen et al., 2026a), *SD3.5-Medium* (Esser et al., 2024), *FLUX.1 [dev]* (Black Forest Labs, 2024), and *LongCat-Image* (Team et al., 2025). We further evaluate whether APT generalizes across model architectures by applying it to the autoregressive image generator *Janus-Pro* (Chen et al., 2025). We use the same APT agentic workflow across all models, without introducing any model-specific prompt sets, reward designs, or hyperparameter searches.

Benchmarks and Metrics. We evaluate on five widely used text-to-image benchmarks that jointly cover compositional generation, prompt following, and text rendering: GenEval (Ghosh et al., 2023), GenEval2 (Kamath et al., 2025), OneIG-EN (Chang et al., 2026), DPG (Hu et al., 2024), and CVTG-2K (Du et al., 2025). Details of each benchmark are provided in Section C. Since the benchmark scores are reported on different scales, we additionally report a normalized average by rescaling GenEval, OneIG-EN, and CVTG-2K to the $[0, 100]$ range and then averaging the scores across all five benchmarks. Crucially, **none of these benchmarks is used to construct the training prompts**: a single APT run produces one post-trained model, which is then evaluated on all five benchmarks, directly testing the benchmark-agnostic nature of APT described in Section 2.4.

Implementation Details. We use DiffusionNFT (Zheng et al., 2026) as our default RL algorithm for post-training. All models are post-trained with LoRA (rank 64, $\alpha = 128$), using a learning rate of $3\text{e-}4$. During rollout, Lens, SD3.5-M, and LongCat-Image use 10 sampling steps, while FLUX.1 [dev] uses 6 steps. The training resolution is 768×768 for Lens and SD3.5-M, and 512×512 for FLUX.1 [dev] and LongCat-Image. For APT, each training round consists of 20 optimization steps. At the beginning of each round, the Diagnosis Agent samples 120 prompt-rubrics pairs from the evaluation set generated by the Evaluation Set Generation Agent and produces an evaluation report. This report is then fed into the RL Data Generation Agent, which automatically generates 480 RL training prompts for the subsequent training round. For the handcrafted RL baseline, we use the same total number of training prompts as APT and follow the same rubric-generation and reward-

Table 1: Comparison of APT with the original base models and a handcrafted RL post-training baseline across four T2I models. The handcrafted RL set is manually curated by human experts to cover diverse generation scenarios, while APT uses the same model-agnostic agentic workflow for all models. APT consistently achieves the best average performance and improves multiple benchmarks with a single post-training run, demonstrating both stronger post-training effectiveness and benchmark-agnostic generalization. Detailed results for each benchmark are shown in Tables 7–10 in the Appendix C.

Model	GenEval	OneIG-EN	GenEval2	DPG	CVTG-2K	Average
	Overall	Average	Soft-TIFA$_{\mathrm{GM}}$	Overall	Word Accuracy	
Lens						
Base Model	0.702	0.521	29.06	85.93	0.480	57.06
w/ Handcrafted RL Set	0.815	0.518	46.35	87.60	0.758	<u>68.61</u>
w/ APT	0.859	0.523	50.73	87.88	0.795	**71.26**
SD3.5-Medium						
Base Model	0.684	0.408	17.63	83.67	0.312	48.34
w/ Handcrafted RL Set	0.722	0.399	27.64	86.54	0.534	<u>55.94</u>
w/ APT	0.763	0.404	29.50	86.50	0.569	**57.92**
FLUX.1 [Dev]						
Base Model	0.660	0.388	22.23	82.54	0.260	47.11
w/ Handcrafted RL Set	0.721	0.435	29.47	85.53	0.310	<u>52.32</u>
w/ APT	0.751	0.459	31.01	85.75	0.513	**57.81**
LongCat-Image						
Base Model	0.727	0.500	28.60	85.71	0.426	55.92
w/ Handcrafted RL Set	0.780	0.503	31.14	87.80	0.730	<u>64.05</u>
w/ APT	0.832	0.510	34.43	87.89	0.821	**67.72**

evaluation procedures. Unless otherwise stated, FLUX.1 [dev] is post-trained for 200 optimization steps, while all other models are post-trained for 400 steps. At inference time, all models are evaluated at a resolution of 768×768 with the following settings: Lens (20 steps, CFG $= 5.0$), SD3.5-M (20 steps, CFG $= 4.5$), FLUX.1 [dev] (16 steps, CFG $= 3.5$), and LongCat-Image (20 steps, CFG $= 4.5$).

3.2 Main Results

In Table 1, we apply APT to four different base models and compare each APT-trained model against two counterparts: its original base model and a version post-trained on a recent *handcrafted* RL training set proposed by Lens (Chen et al., 2026a) and manually curated by human experts. Importantly, all APT experiments use the same agentic workflow, without any model-specific prompt sets, reward designs, or hyperparameter tuning. We draw two main conclusions:

- **APT Consistently Outperforms a Carefully Handcrafted RL Training Set Across All Four Base Models.** The handcrafted RL set is carefully curated by human experts to cover a broad range of generation scenarios, making it a strong post-training baseline. Nevertheless, APT consistently achieves higher average performance across all four models. This suggests that manually designing RL training sets may not be necessary: human experts may struggle to anticipate all relevant failure modes or fully characterize the model-specific output distribution, whereas APT can automatically diagnose each model and adapt its training data accordingly.
- **A Single APT Run Yields Consistent Improvements Across Multiple Benchmarks.** Without using benchmark-specific training prompts or tuning, the same post-trained model improves over its base model across most evaluated benchmarks. This demonstrates strong benchmark-agnostic generalization, showing that APT does not rely on optimization toward any particular benchmark.

3.3 Ablation Study

APT Captures Model-Specific Training Distributions. We investigate whether different T2I models exhibit distinct distributions, and whether APT can automatically adapt its RL data generation to

Table 2: **APT Captures Model-Specific Training Distributions.** Each row corresponds to a target model post-trained with RL data generated by APT from different source models. Diagonal entries denote the standard APT setting, where the RL data is adaptively generated for the same target model, while off-diagonal entries reuse data generated for another model. Detailed results for each benchmark are shown in Tables 11 in the Appendix C.

(a) GenEval

Target/Source	Lens	SD3.5	FLUX	LongCat
Lens	**0.851**	0.833	0.846	0.798
SD3.5	0.736	**0.743**	0.742	0.722
FLUX	0.679	0.726	**0.751**	0.729
LongCat	0.823	0.786	0.827	**0.828**

(b) OneIG-EN

Target/Source	Lens	SD3.5	FLUX	LongCat
Lens	**0.523**	0.522	0.517	0.520
SD3.5	0.404	0.404	**0.405**	0.395
FLUX	0.436	0.443	**0.459**	0.440
LongCat	**0.509**	0.496	0.508	0.506

(c) GenEval2

Target/Source	Lens	SD3.5	FLUX	LongCat
Lens	**46.80**	44.59	46.35	44.98
SD3.5	23.61	**27.58**	23.74	22.65
FLUX	30.06	28.69	**31.01**	30.02
LongCat	34.91	37.37	34.90	**37.80**

(d) Average Over Three Benchmarks

Target/Source	Lens	SD3.5	FLUX	LongCat
Lens	**61.40**	60.03	60.88	58.93
SD3.5	45.87	**47.43**	46.15	44.78
FLUX	47.19	48.53	**50.67**	48.97
LongCat	56.04	55.19	56.13	**57.07**

Table 3: **Ablation of the evaluation set generation agent.** We compare the APT-generated evaluation set with existing benchmark evaluation sets for producing diagnosis reports during post-training. Detailed results for each benchmark are shown in Tables 12–13 in the Appendix C.

Model	GenEval	OneIG-EN	GenEval2	DPG	CVTG-2K	Average
	Overall	Average	Soft-TIFA$_{GM}$	Overall	Word Accuracy	
Lens						
Base Model	0.702	0.521	29.06	85.93	0.480	57.06
OneIG-EN Evaluation Set	0.847	0.528	47.08	87.39	0.823	<u>70.85</u>
GenEval Evaluation Set	0.887	0.505	41.43	84.23	0.674	66.45
APT Evaluation Set	0.859	0.523	50.73	87.88	0.795	**71.26**
SD3.5-Medium						
Base Model	0.684	0.408	17.63	83.67	0.312	48.34
OneIG-EN Evaluation Set	0.800	0.431	26.38	86.69	0.497	<u>57.17</u>
GenEval Evaluation Set	0.692	0.390	17.35	79.02	0.458	50.07
APT Evaluation Set	0.763	0.404	29.50	86.50	0.569	**57.92**

each model. For each of the four models, we compare the standard APT setting against RL using the training data accumulated by APT when post-training another model. As shown in Table 2, each row corresponds to a target model being post-trained with RL data generated for different source models, while the diagonal entry represents the standard APT setting, where the RL data is generated adaptively for the same target model. The diagonal setting achieves the best average performance within every row across all four models. These results indicate that different T2I models exhibit distinct distributions, making RL data generated for one model suboptimal for another. APT addresses this issue by continuously diagnosing the current model and automatically generating RL data tailored to its evolving distribution, rather than relying on a fixed training set shared across models. Note that, for efficiency, we post-train each model for 200 steps in this study.

Ablation on the Evaluation Set Generation Agent. We study the role of the Evaluation Set Generation Agent by replacing the automatically generated APT evaluation set with existing benchmark evaluation sets when producing the diagnosis report. Specifically, we use the OneIG-EN evaluation set and the GenEval evaluation set as alternatives, while keeping the remaining APT pipeline and training configuration unchanged. As shown in Table 3, using the APT-generated evaluation set consistently yields the best average performance on both Lens and SD3.5-Medium. In contrast, relying on a single existing benchmark tends to *bias* the diagnosis toward the corresponding capability and leads to less balanced improvements across benchmarks. This result highlights the importance of automatically constructing a diverse and comprehensive evaluation set for reliable diagnosis and adaptive post-training.

Table 4: **Ablation on the Iterative Feedback Loop and the RL Data Generation Agent.** This study is conducted using the Lens base model. Detailed results for each benchmark are shown in Tables 14–15 in the Appendix C.

Setting	GenEval	OneIG-EN	GenEval2	DPG	CVTG-2K	Average
	Overall	Average	Soft-TIFA$_{\mathrm{GM}}$	Overall	Word Accuracy	
APT	**0.859**	0.523	**50.73**	**87.88**	**0.795**	**71.26**
Ablation on the Iterative Feedback Loop						
w/o Iterative Feedback Loop	0.844	0.523	48.27	87.61	0.791	70.34
Ablation on the RL Data Generation Agent						
Reduced Diagnostic Criteria	0.800	0.520	49.58	87.53	0.763	69.08
Hard-Only Sampling	0.853	**0.530**	47.91	87.58	0.794	70.64

Table 5: **Architecture Generalization to Autoregressive Generation.** Comparison of APT with the original base model and a handcrafted RL post-training baseline on the AR model **Janus-Pro**.

Model	GenEval							OneIG-EN						GenEval2
	Single	Two	Count	Color	Position	Attribute	**Overall**	Alignment	Text	Reasoning	Style	Diversity	**Average**	**Soft-TIFA$_{\mathrm{GM}}$**
Janus-Pro														
Base Model	0.994	0.869	0.538	0.891	0.758	0.635	0.781	0.799	0.068	0.165	0.293	0.243	0.314	15.25
w/ Handcrafted RL Set	0.984	0.891	0.609	0.902	0.790	0.660	<u>0.806</u>	0.804	0.074	0.175	0.297	0.229	<u>0.316</u>	<u>20.42</u>
w/ APT	0.991	0.909	0.606	0.910	0.808	0.668	**0.815**	0.809	0.074	0.175	0.297	0.233	**0.318**	**21.61**

Ablation on the Iterative Feedback Loop. We study whether repeatedly refreshing model feedback is necessary for effective post-training. We compare APT with a variant that continues generating new training prompts but does not re-diagnose the updated model after each round, while keeping the total number of training prompts and RL optimization steps unchanged. As shown in Table 4, removing the iterative feedback loop consistently degrades performance, demonstrating the benefit of continuously adapting the training distribution to the evolving model.

Ablation on the RL Data Generation Agent. We further ablate the prompt-sampling strategy of the RL Data Generation Agent. Removing part of the diagnostic criteria leads to a clear performance drop, while sampling only from the most difficult capabilities is also inferior to the full strategy. These results show that effective RL data generation benefits from both comprehensive diagnostic feedback and a balanced distribution that prioritizes weak capabilities while retaining moderate and mastered ones. The results are shown in Table 4, and the detailed instruction prompts for the two ablation settings are provided in Appendix E.5 and Appendix E.6.

Architecture Generalization to Autoregressive Generation. We further evaluate APT on an autoregressive image generation model, *Janus-Pro* (Chen et al., 2025), using GRPO (Shao et al., 2024) for RL post-training. As shown in Table 5, APT consistently outperforms both the original base model and the handcrafted RL post-training baseline across the evaluated benchmarks. These results demonstrate that APT is not tied to diffusion-based generators and can generalize effectively to autoregressive generation architectures, highlighting its architecture-agnostic nature.

Robustness to Different LLM Backbones. We further examine the sensitivity of APT to the LLM backbone used by the RL Data Generation Agent. All tested backbones yield substantial gains in average performance on *Lens*, suggesting that APT is robust across the evaluated LLM families and scales. Detailed results are provided in Appendix B, Table 6.

4 Conclusion

We presented APT, a fully automated framework for adaptive post-training of text-to-image models. By coordinating evaluation set generation, model diagnosis, and RL data generation, APT adapts the training distribution to each model's evolving capabilities. Experiments across multiple models and benchmarks show higher average performance than post-training on handcrafted RL data, while transfer studies and ablations support the value of model-specific training data and iterative feedback. These findings highlight feedback-driven task adaptation as a practical direction for automated post-training and a step toward recursive self-improvement in text-to-image generation.

AI USE STATEMENT

In this work, we used generative AI tools to assist with manuscript writing and revision. As part of our framework, we used GPT-4.1 to generate evaluation prompts, GPT-4.1-mini to generate evaluation rubrics and assess generated images, and GPT-5.4 to analyze model failures and generate reinforcement learning training prompts, as described in Sections 2.1–2.3. The authors reviewed and revised the AI-assisted manuscript text and take full responsibility for the final content of this work, including its text, claims, and artifacts.

ETHICS STATEMENT

Text-to-image models may generate harmful or misleading imagery. These risks are already present in the underlying base models, and APT itself does not introduce new types of safety risk. We encourage users to apply APT responsibly, remain mindful of these existing risks, and adopt appropriate safeguards. All authors have read and adhere to the ICLR Code of Ethics.

REPRODUCIBILITY STATEMENT

We describe the evaluation-set construction, model diagnosis, and adaptive training-data generation procedures in Sections 2.1–2.3. Section 3.1 specifies the base models, benchmarks, and training and inference settings. The prompts used for evaluation-set construction, rubric generation, reward computation, and adaptive training-data generation are provided in Appendix E.1–E.4. Appendix C provides benchmark descriptions and detailed results, while Appendix B reports additional ablation studies.

REFERENCES

Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. In *International Conference on Learning Representations*, volume 2024, pp. 4965–4987, 2024.

Black Forest Labs. Flux. `https://github.com/black-forest-labs/flux`, 2024.

Black Forest Labs. FLUX.2: Frontier Visual Intelligence. `https://bfl.ai/blog/flux-2`, 2025.

Jingjing Chang, Yixiao Fang, Peng Xing, Shuhan Wu, Wei Cheng, Rui Wang, Xianfang Zeng, Gang Yu, and Hai-Bao Chen. Oneig-bench: Omni-dimensional nuanced evaluation for image generation. *Advances in Neural Information Processing Systems*, 38, 2026.

Dong Chen, Fangyun Wei, Ziyu Wan, Dongdong Chen, Jiawei Zhang, Jinjing Zhao, Sirui Zhang, Yang Yue, Zhiyang Liang, Baining Guo, et al. Lens: Rethinking training efficiency for foundational text-to-image models. *arXiv preprint arXiv:2605.21573*, 2026a.

Jiahui Chen, Philippe Hansen-Estruch, Xiaochuang Han, Yushi Hu, Emily Dinan, Amita Kamath, Michal Drozdzal, Reyhane Askari-Hemmat, Luke Zettlemoyer, and Marjan Ghazvininejad. Unified text-image generation with weakness-targeted post-training. *arXiv preprint arXiv:2601.04339*, 2026b.

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

Nikai Du, Zhennan Chen, Zhizhou Chen, Shan Gao, Xi Chen, Zhengkai Jiang, Jian Yang, and Ying Tai. Textcrafter: Accurately rendering multiple texts in complex visual scenes. *arXiv e-prints*, pp. arXiv–2503, 2025.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024.

Kaituo Feng, Manyuan Zhang, Shuang Chen, Yunlong Lin, Kaixuan Fan, Yilei Jiang, Hongyu Li, Dian Zheng, Chenyang Wang, and Xiangyu Yue. Gen-searcher: Reinforcing agentic search for image generation. *arXiv preprint arXiv:2603.28767*, 2026a.

Ruoyu Feng, Jinming Liu, Yuqi Wang, Xin Cheng, Boyuan Liu, Shanglin Li, Hanshen Zhu, Wenfeng Lin, Mingyu Guo, and Xin Jin. Sefi-image: A text-to-image foundation model with semantic-first diffusion. *arXiv e-prints*, pp. arXiv–2606, 2026b.

Xuelu Feng, Yunsheng Li, Ziyu Wan, Zixuan Gao, Junsong Yuan, Dongdong Chen, and Chunming Qiao. Rubricrl: Simple generalizable rewards for text-to-image generation. In *European Conference on Computer Vision*, pp. 131–149. Springer, 2026c.

Zigang Geng, Yibing Wang, Yeyao Ma, Chen Li, Yongming Rao, Shuyang Gu, Zhao Zhong, Qinglin Lu, Han Hu, Xiaosong Zhang, et al. X-omni: Reinforcement learning makes discrete autoregressive image generative models great again. *arXiv preprint arXiv:2507.22058*, 2025.

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

Google. Gemini 3.1 flash image. `https://deepmind.google/models/model-cards/gemini-3-1-flash-image/`, 2026.

Jun He, Junyan Ye, Zilong Huang, Dongzhi Jiang, Chenjue Zhang, Leqi Zhu, Renrui Zhang, Xiang Zhang, and Weijia Li. Mind-brush: Integrating agentic cognitive search and reasoning into image generation. *arXiv preprint arXiv:2602.01756*, 2026.

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

Chengsong Huang, Wenhao Yu, Xiaoyang Wang, Hongming Zhang, Zongxia Li, Ruosen Li, Jiaxin Huang, Haitao Mi, and Dong Yu. R-zero: Self-evolving reasoning llm from zero data. In *International Conference on Learning Representations*, volume 2026, pp. 130770–130790, 2026.

Dengyang Jiang, Dongyang Liu, Zanyi Wang, Qilong Wu, Liuzhuozheng Li, Hengzhuang Li, Xin Jin, Changsheng Lu, Zhen Li, Mengmeng Wang, et al. Distribution matching distillation meets reinforcement learning. In *European Conference on Computer Vision*, pp. 281–299. Springer, 2026a.

Kaixun Jiang, Yuzheng Wang, Junjie Zhou, Pandeng Li, Zhihang Liu, Chen-Wei Xie, Zhaoyu Chen, Yun Zheng, and Wenqiang Zhang. Genagent: Scaling text-to-image generation via agentic multimodal reasoning. *arXiv preprint arXiv:2601.18543*, 2026b.

Amita Kamath, Kai-Wei Chang, Ranjay Krishna, Luke Zettlemoyer, Yushi Hu, and Marjan Ghazvininejad. Geneval 2: Addressing benchmark drift in text-to-image evaluation. *arXiv preprint arXiv:2512.16853*, 2025.

Baoteng Li, Xianghao Zang, Xinran Wang, Xiangyu Na, Zhixiang He, Hao Sun, Chi Zhang, Zhongjiang He, Tianwei Cao, Kongming Liang, et al. Curriculum group policy optimization: Adaptive sampling for unleashing the potential of text-to-image generation. *arXiv preprint arXiv:2605.17807*, 2026.

Junzhe Li, Yutao Cui, Tao Huang, Weijie Kong, Yiming Cheng, Chuxuan Zeng, Yinping Ma, Chun Fan, Miles Yang, Zhao Zhong, et al. Mixgrpo: Unlocking flow-based grpo efficiency with mixed ode-sde. *arXiv preprint arXiv:2507.21802*, 2025.

Zhanhao Liang, Yuhui Yuan, Shuyang Gu, Bohan Chen, Tiankai Hang, Ji Li, and Liang Zheng. Step-aware preference optimization: Aligning preference with denoising performance at each step. *arXiv preprint arXiv:2406.04314*, 2(5):7, 2024.

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. *Advances in neural information processing systems*, 38:40783–40818, 2026a.

Zunxu Liu, Zhaofan Qiu, Yazhen Xie, Yingwei Pan, Ting Yao, and Tao Mei. On-policy diffusion reinforcement learning meets off-policy quality anchoring. In *European Conference on Computer Vision*, pp. 567–585. Springer, 2026b.

OpenAI. Introducing chatgpt images 2.0. https://openai.com/index/introducing-chatgpt-images-2-0/, 2026.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023.

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

SeedDance. Beyond generation, it understands design — introducing seedream 5.0 pro. https://seed.bytedance.com/en/blog, 2026.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Peize Sun, Yi Jiang, Shoufa Chen, Shilong Zhang, Bingyue Peng, Ping Luo, and Zehuan Yuan. Autoregressive model beats diffusion: Llama for scalable image generation. *arXiv preprint arXiv:2406.06525*, 2024.

Meituan LongCat Team, Hanghang Ma, Haoxian Tan, Jiale Huang, Junqiang Wu, Jun-Yan He, Lishuai Gao, Songlin Xiao, Xiaoming Wei, Xiaoqi Ma, et al. Longcat-image technical report. *arXiv preprint arXiv:2512.07584*, 2025.

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

Kaishen Wang, Ruibo Chen, Tong Zheng, and Heng Huang. Imagent: A unified multimodal agent framework for test-time scalable image generation. *arXiv preprint arXiv:2511.11483*, 2025.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256, 1992.

Yixian Xu, Kaiyuan Gao, Yuxiang Chen, Yilei Chen, Zecheng Tang, Zihao Liu, Zikai Zhou, Deqing Li, Hao Meng, Kuan Cao, et al. Qwen-image-2.0-rl technical report. *arXiv preprint arXiv:2606.27608*, 2026.

Mingde Yao, Zhiyuan You, King-Man Tam, Menglu Wang, and Tianfan Xue. Photoagent: Agentic photo editing with exploratory visual aesthetic planning. *arXiv e-prints*, pp. arXiv–2602, 2026.

Eric Zelikman, Eliana Lorch, Lester Mackey, and Adam Tauman Kalai. Self-taught optimizer (stop): Recursively self-improving code generation. *arXiv preprint arXiv:2310.02304*, 2023.

Xinjie Zhang, Peng Zhang, Shicheng Zheng, Jinghao Guo, Zhaoyang Jia, Yifei Shen, Xun Guo, Yuxuan Luo, Jiahao Li, Wenxuan Xie, et al. Mage-flow: An efficient native-resolution foundation model for image generation and editing. *arXiv preprint arXiv:2607.19064*, 2026a.

Zekai Zhang, Jiahao Li, Jie Zhang, Kaiyuan Gao, Kun Yan, Lihan Jiang, Ningyuan Tang, Shengming Yin, Tianhe Wu, Xiaoyue Chen, et al. Qwen-image-agent: Bridging the context gap in real-world image generation. *arXiv preprint arXiv:2606.26907*, 2026b.

Bing Zhao, Chenfei Wu, Deqing Li, Hao Meng, Jiahao Li, Jie Zhang, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kuan Cao, et al. Qwen-image-2.0 technical report. *arXiv preprint arXiv:2605.10730*, 2026.

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*, volume 2026, pp. 134129–134150, 2026.

Adam Zweiger, Jyo Pari, Han Guo, Yoon Kim, and Pulkit Agrawal. Self-adapting language models. *Advances in Neural Information Processing Systems*, 38:74084–74115, 2026.

A Related Work

Post-Training for Text-to-Image Generation. Post-training has become a standard paradigm for adapting pre-trained T2I models to downstream objectives. Supervised fine-tuning (SFT) adapts a model to curated image–text pairs, with performance gains largely determined by data filtering, recaptioning, and aesthetic curation, but it remains bounded by the collected corpus and costly human curation. A complementary line directly optimizes downstream rewards following the policy-gradient view of reinforcement learning (Williams, 1992; Schulman et al., 2017): DDPO (Black et al., 2024) formulates denoising as a multi-step decision-making problem and optimizes diffusion models with policy gradients; Diffusion-DPO (Wallace et al., 2024) adapts direct preference optimization (Rafailov et al., 2023) to diffusion models, and SPO (Liang et al., 2024) refines it with step-aware preference signals; more recently, Flow-GRPO (Liu et al., 2026a) and follow-ups (Li et al., 2025; Liu et al., 2026b) extend online policy-gradient training to flow matching via stochastic sampling, and DiffusionNFT (Zheng et al., 2026) performs negative-aware fine-tuning on the forward process without likelihood estimation.

Agentic Image Generation. Recent work augments image generation and editing with agentic capabilities such as planning (Yao et al., 2026), reasoning over implicit user intent (He et al., 2026), web and image search for open-world grounding (Feng et al., 2026a), and self-feedback for test-time scaling (Jiang et al., 2026b; Wang et al., 2025), with emerging systems unifying these abilities in a single framework (Zhang et al., 2026b).

These methods share a common design: *the agent operates at inference time around a frozen generator*, improving how a fixed model is used rather than changing the model itself. As a result, intrinsic failure modes persist once the agentic scaffolding is removed, and every deployment pays the extra inference cost. APT instead moves the agent from the *inference* loop into the *post-training* loop: discover model weaknesses through evaluation and address them through training. The gains are absorbed into the model parameters and retained under standard single-pass inference, and the training distribution is re-derived from the current model at every round, making APT *model-adaptive* rather than applying one fixed agentic policy to any generator. The two directions are thus complementary: prior work makes better use of a given model, while APT makes the model itself better.

B More Ablation Studies

Robustness to Different LLM Backbones in the RL Data Generation Agent. We investigate how sensitive APT is to the choice of LLM backbone used by the RL Data Generation Agent. Using *Lens* as the base image generation model, we replace the default GPT-5.4 backbone with LLMs from different families and model scales, while keeping all other components of the post-training pipeline fixed. As shown in Table 6, all tested LLM backbones lead to effective post-training and achieve comparable improvements across the five benchmarks, despite noticeable differences in their absolute performance. This suggests that the effectiveness of APT does not depend on a specific LLM family or scale, and that the RL Data Generation Agent remains robust across diverse proprietary and open-source backbones.

C Detailed Benchmark Results

We evaluate APT on five benchmarks, with the details of each benchmark described as follows:

- **GenEval** (Ghosh et al., 2023) focuses on object-centric compositional alignment, with 553 prompts spanning six templated tasks: single-object generation, two-object co-occurrence, counting, color, spatial relation, and attribute binding. It uses detector- and classifier-based verification to assess whether generated images satisfy these structured constraints.
- **OneIG-EN** (Chang et al., 2026) is the English split of OneIG-Bench, comprising 1,120 prompts across general objects, portraits, anime/stylization, text rendering, and knowledge/reasoning. It evaluates generated images with dimension-specific scores for subject alignment, text accuracy, reasoning, style, diversity, and overall performance.
- **GenEval2** (Kamath et al., 2025) extends GenEval with 800 prompts of increasing compositionality, covering 40 objects, 18 attributes, 9 relations, and counts from 2 to 7. Each prompt

Table 6: Ablation on the LLM backbone used in the RL Data Generation Agent. We replace the default GPT-5.4 with LLMs from different families and scales while keeping the remaining APT pipeline unchanged. Results are reported on five benchmarks using *Lens* as the base model.

Model	GenEval	OneIG-EN	GenEval2	DPG	CVTG-2K	Average
	Overall	Average	Soft-TIFA$_{\mathrm{GM}}$	Overall	Word Accuracy	
Base Model	0.702	0.521	29.06	85.93	0.480	57.06
GPT Series						
GPT-5.4	0.859	0.523	50.73	87.88	0.795	71.26
GPT-5.6-Sol	0.832	0.519	51.43	88.12	0.791	70.75
GPT-4.1	0.851	0.520	48.94	87.93	0.796	70.71
GPT-5.4-mini	0.825	0.518	50.79	87.69	0.810	70.76
Qwen Series						
Qwen3.5-9B	0.861	0.520	47.03	88.36	0.793	70.56
Qwen3.5-122B-A10B	0.893	0.525	51.42	87.65	0.805	72.27
Gemini Series						
Gemini-3.8-Flash	0.829	0.525	48.67	87.70	0.781	69.97
Grok Series						
Grok-4.6	0.819	0.501	51.44	87.50	0.732	68.83

contains 3 to 10 visual atoms, including objects, attributes, counts, and relations, and is evaluated using Soft-TIFA, which decomposes prompts into atom-level visual questions and aggregates soft correctness scores to measure fine-grained compositional alignment. Soft-TIFA$_{\mathrm{AM}}$ averages these scores arithmetically within each image to estimate atom-level correctness, whereas Soft-TIFA$_{\mathrm{GM}}$ uses their geometric mean to capture prompt-level correctness by more strongly penalizing poorly satisfied atoms. Both variants then average the image-level scores arithmetically across the benchmark. We report Soft-TIFA$_{\mathrm{GM}}$, as the GenEval2 (Kamath et al., 2025) finds that it achieves higher AUROC against prompt-level human judgments than Soft-TIFA$_{\mathrm{AM}}$ across the original, rewritten, and combined prompt sets.

- **DPG** (Hu et al., 2024) evaluates dense prompt following with 1,065 lengthy prompts describing multiple objects, fine-grained attributes, and complex relationships. It decomposes each prompt into a dependency-aware question graph and employs an MLLM-based VQA evaluator to measure alignment across global, entity, attribute, relation, and other semantic dimensions.
- **CVTG-2K** (Du et al., 2025) evaluates complex visual text generation with multiple text regions, where prompts vary the number of regions from 2 to 5 and specify text content, position, length, and style attributes such as color, font, and size.

Table 7 provides the detailed **GenEval** results corresponding to Table 1.

Table 8 provides the detailed **OneIG-EN** results corresponding to Table 1. We note that RL post-training can substantially reduce generation diversity, a phenomenon observed for both the Lens (Chen et al., 2026a) and Mage-Flow (Zhang et al., 2026a) models. Consequently, the **Diversity** score on OneIG-EN decreases markedly after RL post-training and may offset improvements in other dimensions, occasionally resulting in an average score lower than that of the base model. Nevertheless, the post-trained models consistently exhibit stronger performance in capability-oriented dimensions such as subject alignment, text rendering, and reasoning. More importantly, despite the diversity degradation shared by RL-based approaches, our method *w/* APT generally achieves higher alignment, text, reasoning, and overall scores than *w/* Handcrafted RL Set. These results suggest that APT produces more effective post-training data and improves the target capabilities more effectively than handcrafted RL post-training.

Table 9 provides the detailed **DPG** results corresponding to Table 1.

Table 10 provides the detailed **CVTG-2K** results corresponding to Table 1.

Table 11 provides the detailed **GenEval** and **OneIG-EN** results corresponding to Table 2.

Table 12 provides the detailed **GenEval** and **OneIG-EN** results corresponding to Table 3.

Table 13 provides the detailed **DPG** and **CVTG-2K** results corresponding to Table 3.

Table 14 provides the detailed **GenEval** and **OneIG-EN** results corresponding to Table 4.

Table 15 provides the detailed **DPG** and **CVTG-2K** results corresponding to Table 4.

Table 7: Detailed comparison of APT with the original base models and a handcrafted RL post-training baseline across four T2I models on **GenEval**.

Model	Single	Two	Count	Color	Position	Attribute	**Overall**
Lens							
Base Model	0.991	0.788	0.547	0.848	0.548	0.488	0.702
w/ Handcrafted RL Set	0.994	0.929	0.763	0.848	0.735	0.620	<u>0.815</u>
w/ APT	1.000	0.947	0.822	0.872	0.835	0.680	**0.859**
SD3.5-Medium							
Base Model	0.994	0.854	0.593	0.844	0.260	0.560	0.684
w/ Handcrafted RL Set	0.997	0.955	0.622	0.840	0.370	0.550	<u>0.722</u>
w/ APT	0.997	0.932	0.794	0.862	0.335	0.660	**0.763**
FLUX.1 [Dev]							
Base Model	0.981	0.813	0.709	0.801	0.228	0.428	0.660
w/ Handcrafted RL Set	0.972	0.869	0.813	0.798	0.295	0.580	<u>0.721</u>
w/ APT	0.988	0.924	0.756	0.846	0.353	0.638	**0.751**
LongCat-Image							
Base Model	0.975	0.854	0.575	0.846	0.528	0.583	0.727
w/ Handcrafted RL Set	0.984	0.939	0.569	0.854	0.730	0.605	<u>0.780</u>
w/ APT	0.984	0.960	0.656	0.899	0.780	0.710	**0.832**

Table 8: Detailed comparison of APT with the original base models and a handcrafted RL post-training baseline across four T2I models on **OneIG-EN**.

Model	Alignment	Text	Reasoning	Style	Diversity	**Average**
Lens						
Base Model	0.840	0.872	0.247	0.387	0.257	<u>0.521</u>
w/ Handcrafted RL Set	0.868	0.949	0.274	0.314	0.184	0.518
w/ APT	0.874	0.947	0.284	0.335	0.174	**0.523**
SD3.5-Medium						
Base Model	0.793	0.341	0.288	0.363	0.254	**0.408**
w/ Handcrafted RL Set	0.823	0.379	0.302	0.316	0.173	0.399
w/ APT	0.825	0.410	0.305	0.314	0.164	<u>0.404</u>
FLUX.1 [Dev]						
Base Model	0.780	0.332	0.234	0.352	0.244	0.388
w/ Handcrafted RL Set	0.824	0.548	0.264	0.330	0.208	<u>0.435</u>
w/ APT	0.824	0.655	0.272	0.340	0.203	**0.459**
LongCat-Image						
Base Model	0.816	0.750	0.228	0.396	0.309	0.500
w/ Handcrafted RL Set	0.863	0.877	0.235	0.322	0.220	<u>0.503</u>
w/ APT	0.872	0.907	0.236	0.322	0.215	**0.510**

Table 9: Detailed comparison of APT with the original base models and a handcrafted RL post-training baseline across four T2I models on **DPG**.

Model	Global	Entity	Attribute	Relation	Other	**Overall**
Lens						
Base Model	93.47	90.44	91.05	92.16	90.49	85.93
w/ Handcrafted RL Set	89.88	92.69	91.77	93.98	91.90	<u>87.60</u>
w/ APT	93.46	92.94	92.73	92.46	91.73	**87.88**
SD3.5-Medium						
Base Model	92.15	89.67	88.60	91.89	85.19	83.67
w/ Handcrafted RL Set	90.34	91.98	91.71	92.27	89.96	**86.54**
w/ APT	91.53	93.05	90.82	91.50	90.91	<u>86.50</u>
FLUX.1 [Dev]						
Base Model	87.53	88.41	88.21	89.55	88.75	82.54
w/ Handcrafted RL Set	91.37	91.12	89.83	91.30	87.78	<u>85.53</u>
w/ APT	92.25	91.49	90.61	88.77	90.51	**85.75**
LongCat-Image						
Base Model	85.68	90.39	90.21	91.78	88.68	85.71
w/ Handcrafted RL Set	91.76	93.11	92.24	92.41	93.10	<u>87.80</u>
w/ APT	93.11	93.10	92.17	91.59	92.84	**87.89**

Table 10: Detailed comparison of APT with the original base models and a handcrafted RL post-training baseline across four T2I models on **CVTG-2K**.

Model	Word Accuracy	NED	CLIPScore	VQAScore	Aesthetic
Lens					
Base Model	0.480	0.711	0.785	0.932	4.986
w/ Handcrafted RL Set	0.758	0.878	0.786	0.959	5.926
w/ APT	**0.795**	0.905	0.784	0.963	5.999
SD3.5-Medium					
Base Model	0.312	0.626	0.754	0.891	5.541
w/ Handcrafted RL Set	0.534	0.798	0.783	0.943	5.825
w/ APT	**0.569**	0.837	0.819	0.938	5.604
FLUX.1 [Dev]					
Base Model	0.260	0.499	0.710	0.858	6.050
w/ Handcrafted RL Set	0.310	0.557	0.742	0.915	6.053
w/ APT	**0.513**	0.725	0.777	0.931	6.404
LongCat-Image					
Base Model	0.426	0.682	0.756	0.935	5.501
w/ Handcrafted RL Set	0.730	0.866	0.781	0.960	6.032
w/ APT	**0.821**	0.927	0.775	0.961	6.054

Table 11: Detailed results of **APT Captures Model-Specific Training Distributions** on **GenEval** and **OneIG-EN**. Each block corresponds to a target model post-trained with RL data generated by APT from different source models, while rows denote the source model used for data generation. Shaded rows denote the standard APT setting.

Data Source	GenEval							OneIG-EN					
	Single	Two	Count	Color	Position	Attribute	**Overall**	Alignment	Text	Reasoning	Style	Diversity	**Average**
Target Model: Lens													
Lens	1.000	0.960	0.831	0.872	0.768	0.675	**0.851**	0.871	0.952	0.278	0.338	0.176	**0.523**
SD3.5-Medium	1.000	0.934	0.813	0.862	0.748	0.640	0.833	0.869	0.941	0.277	0.343	0.181	0.522
FLUX.1 [Dev]	0.997	0.939	0.797	0.875	0.813	0.658	0.846	0.871	0.950	0.276	0.317	0.173	0.517
LongCat-Image	0.997	0.917	0.728	0.819	0.733	0.595	0.798	0.865	0.946	0.270	0.321	0.199	0.520
Target Model: SD3.5-Medium													
Lens	0.994	0.904	0.794	0.811	0.318	0.595	0.736	0.821	0.398	0.309	0.316	0.174	0.404
SD3.5-Medium	0.988	0.907	0.781	0.835	0.343	0.605	**0.743**	0.824	0.402	0.307	0.322	0.166	0.404
FLUX.1 [Dev]	0.994	0.904	0.797	0.840	0.335	0.580	0.742	0.820	0.379	0.312	0.325	0.189	**0.405**
LongCat-Image	1.000	0.957	0.550	0.848	0.313	0.665	0.722	0.828	0.383	0.292	0.303	0.171	0.395
Target Model: FLUX.1 [Dev]													
Lens	0.997	0.833	0.709	0.785	0.260	0.488	0.679	0.810	0.558	0.264	0.331	0.215	0.436
SD3.5-Medium	0.988	0.884	0.763	0.827	0.298	0.598	0.726	0.812	0.600	0.262	0.338	0.201	0.443
FLUX.1 [Dev]	0.988	0.924	0.756	0.846	0.353	0.638	**0.751**	0.824	0.655	0.272	0.340	0.203	**0.459**
LongCat-Image	1.000	0.922	0.738	0.817	0.273	0.625	0.729	0.811	0.594	0.269	0.322	0.204	0.440
Target Model: LongCat-Image													
Lens	0.994	0.970	0.644	0.880	0.738	0.710	0.823	0.862	0.907	0.235	0.316	0.224	**0.509**
SD3.5-Medium	0.997	0.955	0.653	0.891	0.603	0.620	0.786	0.860	0.865	0.234	0.309	0.211	0.496
FLUX.1 [Dev]	0.991	0.942	0.763	0.864	0.740	0.660	0.827	0.860	0.882	0.235	0.338	0.224	0.508
LongCat-Image	0.991	0.972	0.688	0.910	0.728	0.683	**0.828**	0.863	0.880	0.237	0.324	0.226	0.506

Table 12: Detailed ablation of the evaluation set generation agent on **GenEval** and **OneIG-EN**.

Model	GenEval							OneIG-EN					
	Single	Two	Count	Color	Position	Attribute	**Overall**	Alignment	Text	Reasoning	Style	Diversity	**Average**
Lens													
Base Model	0.991	0.788	0.547	0.848	0.548	0.488	0.702	0.840	0.872	0.247	0.387	0.257	0.521
OneIG-EN Evaluation Set	0.997	0.929	0.788	0.912	0.770	0.688	0.847	0.869	0.969	0.292	0.344	0.166	**0.528**
GenEval Evaluation Set	0.984	0.957	0.900	0.864	0.930	0.685	**0.887**	0.823	0.908	0.273	0.330	0.193	0.505
APT Evaluation Set	1.000	0.947	0.822	0.872	0.835	0.680	0.859	0.874	0.947	0.284	0.335	0.174	0.523
SD3.5-Medium													
Base Model	0.994	0.854	0.593	0.844	0.260	0.560	0.684	0.793	0.341	0.288	0.363	0.254	0.408
OneIG-EN Evaluation Set	1.000	0.957	0.859	0.880	0.430	0.675	**0.800**	0.832	0.463	0.302	0.378	0.180	**0.431**
GenEval Evaluation Set	0.959	0.745	0.663	0.745	0.680	0.363	0.692	0.693	0.413	0.287	0.330	0.227	0.390
APT Evaluation Set	0.997	0.932	0.794	0.862	0.335	0.660	0.763	0.825	0.410	0.305	0.314	0.164	0.404

Table 13: Detailed ablation of the evaluation set generation agent on **DPG** and **CVTG-2K**.

Model	DPG						CVTG-2K				
	Global	Entity	Attribute	Relation	Other	**Overall**	**Word Accuracy**	NED	CLIPScore	VQAScore	Aesthetic
Lens											
Base Model	93.47	90.44	91.05	92.16	90.49	85.93	0.480	0.711	0.785	0.932	4.986
OneIG-EN Evaluation Set	88.74	93.00	91.94	90.85	93.26	87.39	**0.823**	0.938	0.814	0.961	5.718
GenEval Evaluation Set	88.46	91.78	89.81	87.63	88.77	84.23	0.674	0.852	0.797	0.941	5.182
APT Evaluation Set	93.46	92.94	92.73	92.46	91.73	**87.88**	0.795	0.905	0.784	0.963	5.999
SD3.5-Medium											
Base Model	92.15	89.67	88.60	91.88	85.19	83.67	0.312	0.626	0.754	0.891	5.541
OneIG-EN Evaluation Set	91.38	91.26	91.29	93.36	90.36	**86.69**	0.497	0.771	0.802	0.939	5.722
GenEval Evaluation Set	86.54	85.01	86.68	84.52	86.99	79.02	0.458	0.770	0.780	0.882	4.797
APT Evaluation Set	91.53	93.05	90.82	91.50	90.91	86.50	**0.569**	0.837	0.819	0.938	5.604

Table 14: Detailed ablation of the Iterative Feedback Loop and the RL Data Generation Agent on **GenEval** and **OneIG-EN**.

Setting	GenEval							OneIG-EN					
	Single	Two	Count	Color	Position	Attribute	**Overall**	Alignment	Text	Reasoning	Style	Diversity	**Average**
APT	1.000	0.947	0.822	0.872	0.835	0.680	**0.859**	0.874	0.947	0.284	0.335	0.174	0.523
Ablation on the Iterative Feedback Loop													
w/o Iterative Feedback Loop	0.982	0.955	0.800	0.874	0.778	0.676	0.844	0.866	0.957	0.286	0.328	0.176	0.523
Ablation on the RL Data Generation Agent													
Reduced Diagnostic Criteria	1.000	0.886	0.750	0.878	0.668	0.618	0.800	0.871	0.939	0.281	0.330	0.180	0.520
Hard-Only Sampling	1.000	0.965	0.869	0.825	0.818	0.643	0.853	0.869	0.960	0.292	0.339	0.190	**0.530**

Table 15: Detailed ablation of the Iterative Feedback Loop and the RL Data Generation Agent on **DPG** and **CVTG-2K**.

Setting	DPG						CVTG-2K				
	Global	Entity	Attribute	Relation	Other	**Overall**	**Word Accuracy**	NED	CLIPScore	VQAScore	Aesthetic
APT	93.46	92.94	92.73	92.46	91.73	**87.88**	**0.795**	0.905	0.784	0.963	5.999
Ablation on the Iterative Feedback Loop											
w/o Iterative Feedback Loop	94.46	91.75	92.38	93.49	88.65	87.61	0.791	0.912	0.807	0.962	5.494
Ablation on the RL Data Generation Agent											
Reduced Diagnostic Criteria	93.77	91.74	92.13	89.53	94.54	87.53	0.763	0.900	0.796	0.963	5.769
Hard-Only Sampling	93.71	93.55	91.65	91.87	89.08	87.58	0.794	0.917	0.806	0.962	5.734

D VISUALIZATION

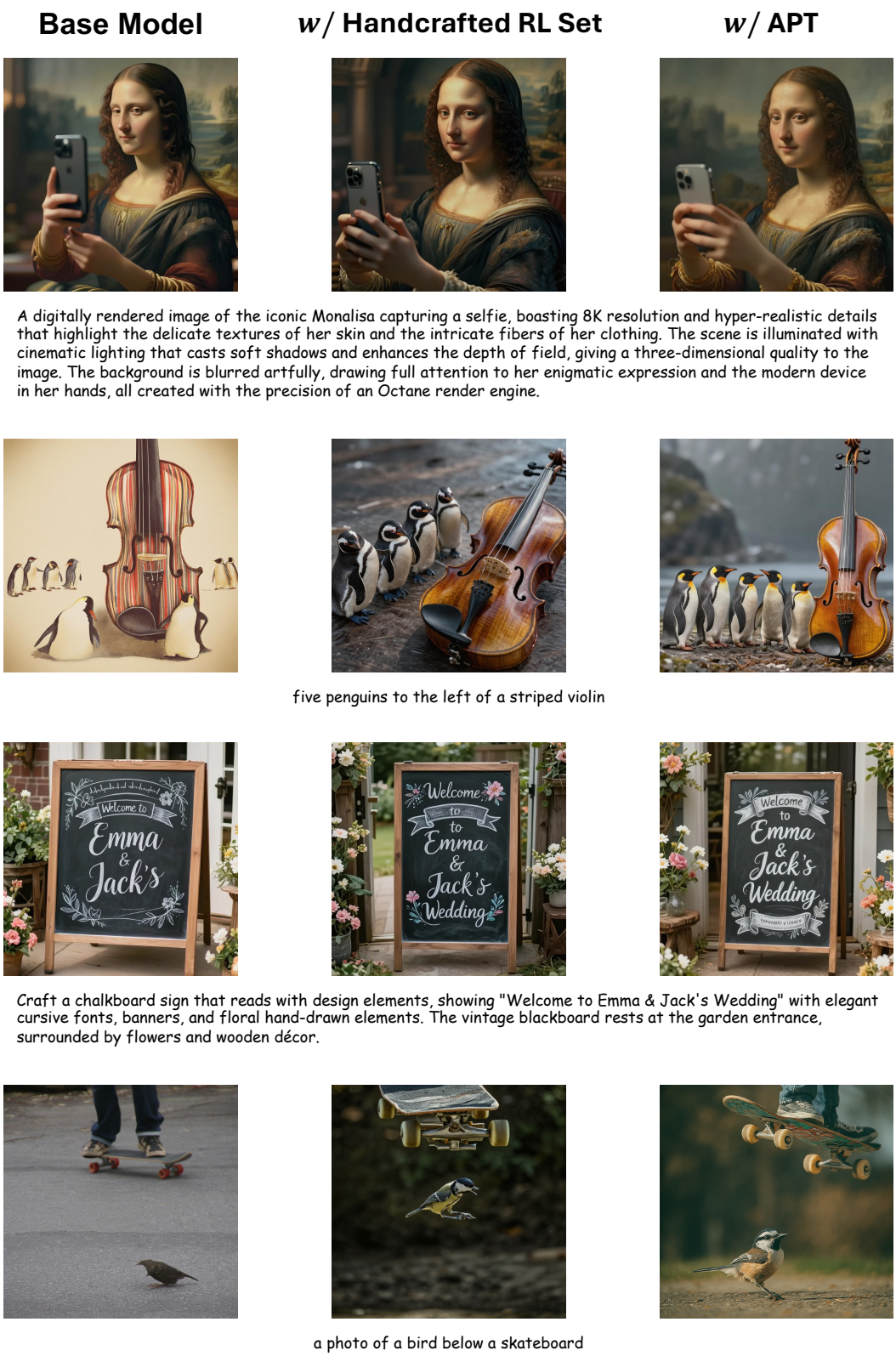


Figure 3: Qualitative comparison of the base model, handcrafted RL baseline, and APT on *Lens*.



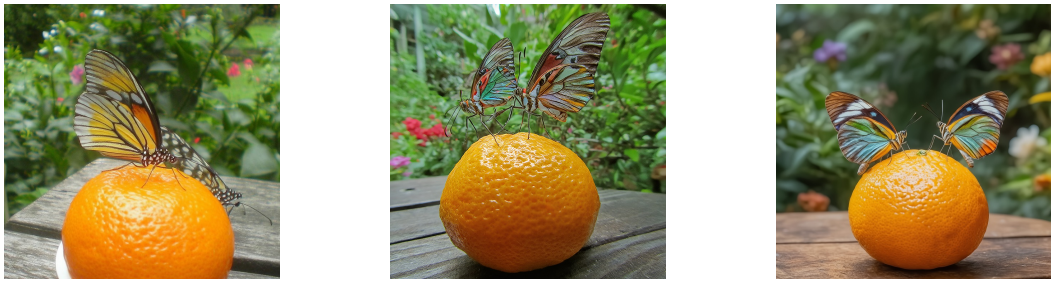
The image captures a whimsical scene with a brown tabby cat, its fur patterned in shades of dark brown, black, and light taupe. The cat, situated as if in the throes of space, is portrayed with a transparent, gleaming bubble encasing its head like an astronaut's helmet. Around it, an assortment of smaller bubbles float serenely in the imagined cosmos, with a creatively interpreted Saturn adorned with rings in the backdrop, providing an aura of interstellar exploration.



a photo of a purple elephant

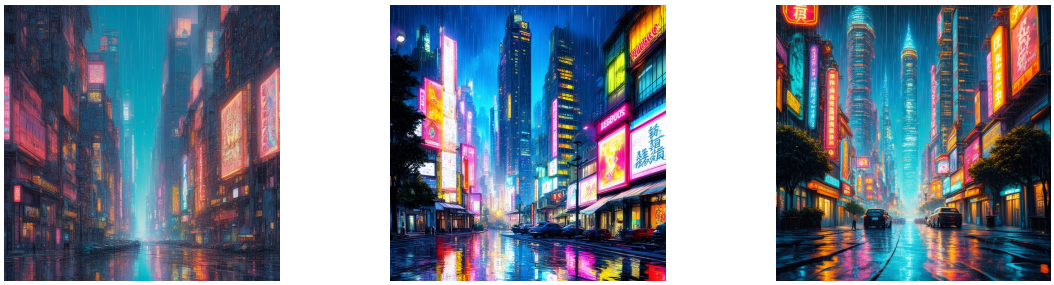


Try crafting a bold street art design spelling "No Limits, No Rules — Just Pure Expression" using thick outlines, vivid spray textures, and abstract shapes on a metal garage door.



Two multicolored butterflies with delicate, veined wings gently balance atop a vibrant, orange tangerine in a bustling garden. The tangerine, with its glossy, dimpled texture, is situated on a wooden table, contrasting with the greenery of the surrounding foliage and flowers. The butterflies, appearing nearly small in comparison, add a touch of grace to the scene, complementing the natural colors of the verdant backdrop.

Figure 4: Additional qualitative comparison of the base model, handcrafted RL baseline, and APT on *Lens*.



a highly intricate and vibrant cityscape that reflects a fusion of Moebius's imaginative design and Makoto Shinkai's detailed animation style. The streets are aglow with neon signs in a kaleidoscope of colors, casting reflections on the glossy, rain-slicked pavements. Towering skyscrapers with glowing windows rise towards a starless night sky, as the artwork garners significant attention and praise on ArtStation..



four plastic cookies in front of a bagel



Craft a chalkboard sign that reads with design elements, showing "Welcome to Emma & Jack's Wedding" with elegant cursive fonts, banners, and floral hand-drawn elements. The vintage blackboard rests at the garden entrance, surrounded by flowers and wooden décor.



Design a heartwarming t-shirt for gecko lovers, centered on a charming gecko hugging a glowing heart. Emphasize the gecko's gentle expression, highlighting the 'warm heart' theme. It should feature five fingers and vibrant, soothing colors to showcase its unique texture and patterns. Keep the background minimalistic, ensuring the focus remains on the gecko and a playful, easy-to-read slogan. Strive for a blend of cuteness and sophistication to attract a wide audience of gecko enthusiasts.

Figure 5: Qualitative comparison of the base model, handcrafted RL baseline, and APT on *SD3.5-Medium*.

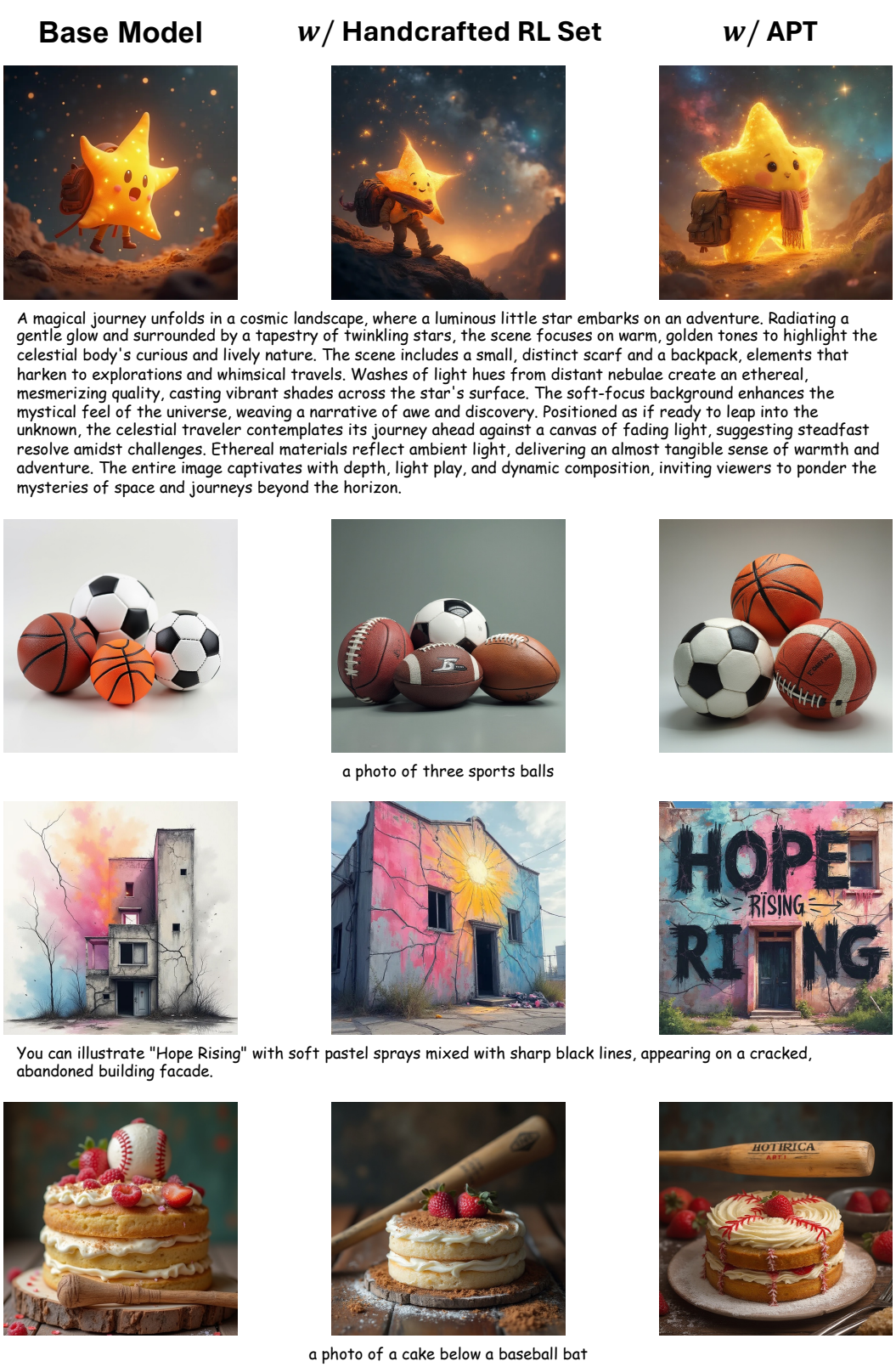


Figure 6: Qualitative comparison of the base model, handcrafted RL baseline, and APT on *FLUX.1[Dev]*.

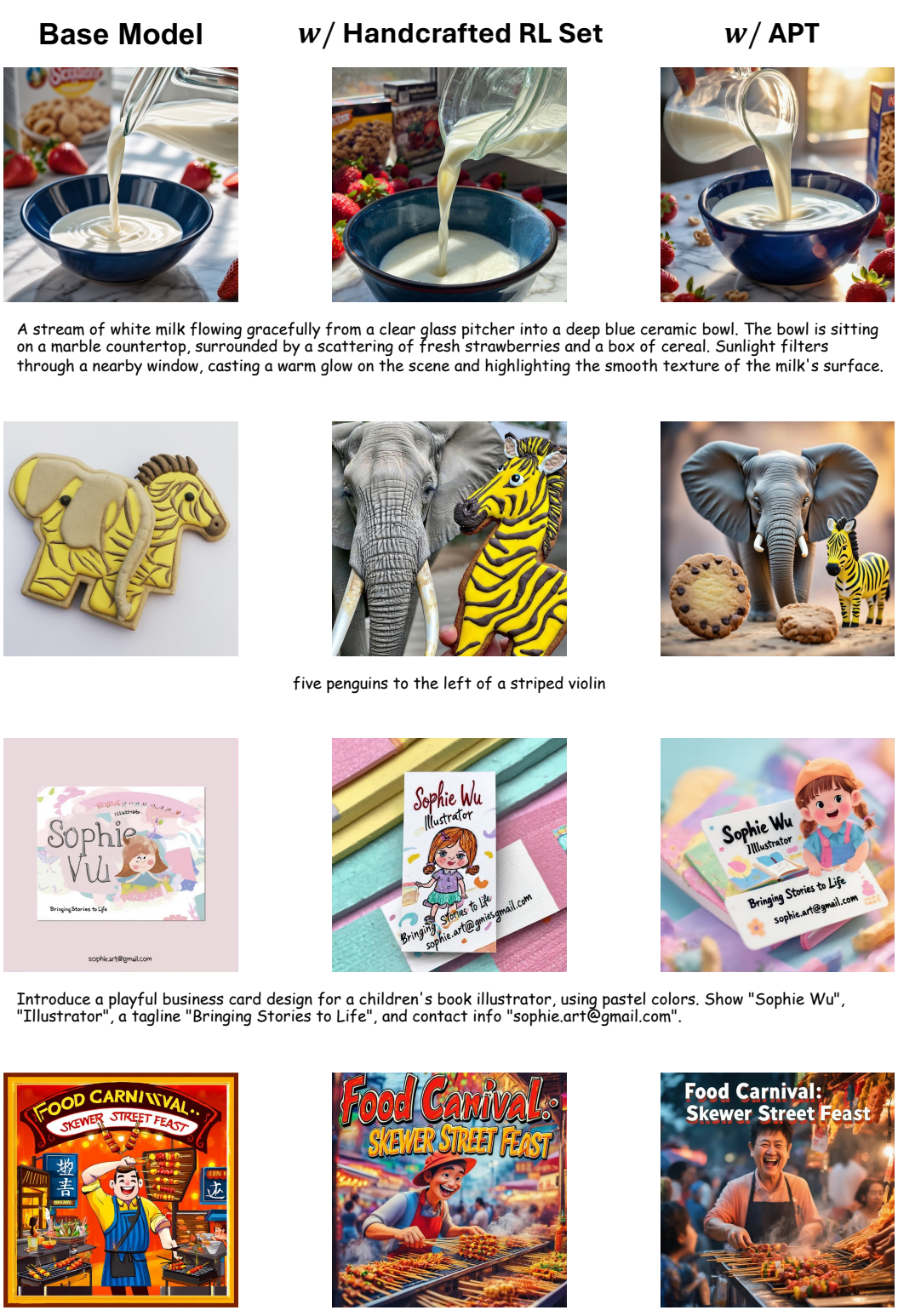


Figure 7: Qualitative comparison of the base model, handcrafted RL baseline, and APT on *LongCat-Image*.

E PROMPT

E.1 PROMPT FOR EVALUATION PROMPT GENERATION

Prompt for Evaluation Prompt Generation

You are an expert prompt engineer for photo-realistic image generation.

Given an entity (a concept, object, person type, scene, etc.) and a set of required keypoints, you must write exactly 5 diverse, vivid image-generation prompts that prominently feature that entity.

Keypoint definitions: When a keypoint is required, you MUST naturally incorporate relevant detail into every prompt:

- **global:** describe the overall scene composition, atmosphere, and environment surrounding the entity.
- **counting:** include specific quantities of objects or subjects (e.g., "three apples", "a pair of shoes").
- **position:** specify spatial placement or pose of the entity (e.g., "in the center", "on the left side", "foreground vs background").
- **attribute:** describe distinctive physical attributes (e.g., shape, size, material, texture, condition).
- **colors:** mention explicit color details for the entity or surrounding elements (e.g., "deep crimson", "pale blue").
- **relation:** describe spatial or semantic relationships between the entity and other objects or people in the scene.
- **style:** specify a particular visual or photographic style (e.g., "cinematic", "macro photography", "vintage film grain").
- **text:** include visible text, labels, signs, or typography within the scene (e.g., a storefront sign, a book title, a label on a bottle).

Rules:

1. Each prompt should be 30–60 words, written as a single descriptive paragraph (no line breaks).
2. Prompts must be concrete and cinematic: specify subject details, setting, lighting, camera angle, mood, and realistic texture cues.
3. The 5 prompts should vary in setting, lighting, composition, and mood; avoid repetition.
4. The entity must be the clear focal subject of every prompt.
5. ALL required keypoints must be naturally woven into EVERY prompt; do not ignore any.
6. Return ONLY a valid JSON object with exactly these keys: "prompt-1", "prompt-2", "prompt-3", "prompt-4", and "prompt-5". No extra keys, no markdown fences, no explanation.

Style and length reference: ONLY use the following examples for tone, structure, and approximate length. Do NOT copy or imitate their specific content, subjects, or settings. Your prompts must be entirely about the given entity.

- "A close-up portrait of a young woman with clear skin, dark almond-shaped eyes, and long black hair, facing the camera directly, neutral expression, soft daylight from a nearby window, natural skin texture, realistic eyelashes, subtle lip color, shallow depth of field, photo-realistic."
- "A middle-aged man with short salt-and-pepper hair and a trimmed beard, sitting in a leather chair in a quiet study, wearing a navy sweater, warm lamp light illuminating one side of his face, detailed wrinkles around the eyes, thoughtful expression, realistic portrait photography."
- "A smiling teenage boy standing outdoors on a school campus, wearing a blue hoodie and backpack, slightly messy hair, bright morning light, candid expression, natural skin detail, realistic ears, hands, and facial proportions."
- "An elderly woman with silver hair tied into a bun, wearing thin round glasses and a beige cardigan, seated by a window with soft afternoon sunlight, expressive wrinkles, gentle smile, realistic age detail in face and hands, warm domestic atmosphere."

- "A young man with curly hair and freckles, standing in front of a plain gray studio background, wearing a white T-shirt, direct eye contact, relaxed posture, even studio lighting, crisp facial detail, realistic pores and skin texture."

E.2 Prompt for Rubric Generation

We adopt the following system prompt to generate rubrics for each prompt.

Prompt for Rubric Generation

You are an expert in evaluating the quality of images generated by text-to-image models. A user will provide a text prompt. Your task is to reason about the prompt and identify the key rubrics that real users would use to evaluate the generated image. You should consider the rubric categories listed below **only if** they are relevant to the prompt.

1. "OCR Alignment: If the text prompt explicitly or implicitly implies or requires text to appear in the image, evaluate whether any textual content (e.g., labels, signs, clothing text) within the generated image matches the OCR text that is implied or stated in the prompt. If the text prompt says no OCR text/specific OCR text, you should have a corresponding rubric as well. If the given text prompt (e.g., "a person walking on the beach") does not contain any OCR text related requirement, please do not return OCR-related rubric."
2. "Object Count Consistency: Include this if the prompt explicitly or implicitly specifies the number of key objects or subjects. Assess whether the number of objects described in the prompt is accurately reflected in the image. If no specific object count is mentioned in the text prompt, please do not return this count-related rubric."
3. "Attribute Accuracy: Include this if the prompt explicitly describes appearance-level attributes (e.g., color, material, texture, patterns). Examine whether each object's visual attributes match the prompt."
4. "Action Accuracy: Include this if the prompt explicitly describes actions or dynamic states (e.g., running, holding, playing). Examine whether the actions and interactions match the prompt."
5. "Object Placement and Spatial Reasoning: Include this if the prompt implies or specifies spatial relations between objects. Evaluate whether the spatial relationships or positioning of objects in the image correspond to what is expected from the prompt (e.g., "the dog is sitting under the table")."

Please output up to 10 rubrics in a single JSON object.

Each key should be a **short rubric name** (e.g., "Object Count Consistency"), and each value should be a **brief explanation**.

Valid example:
{"Object Count Consistency": "Verify that exactly one cat is shown.", "OCR Alignment": "Ensure visible text matches prompt description."}

Invalid example (CSV format):
"Object Count Consistency","Verify that exactly one cat is shown."
Invalid examples (DO NOT DO THESE):
- {"Object counting": "..."} # INVALID renamed base key
- {"Count Object": "..."} # INVALID reordered/changed
- {"Object Count Consistency": "..."} # INVALID spacing changed

Strictly follow this format:
- A single JSON object (not a list)
- Keys and values wrapped in double quotes
- do not change the key name
- No trailing commas or comments
- No line breaks within keys or values

Your output must be valid JSON parsable with Python's "json.loads()".

E.3 Prompt for RL Reward Generation

Given a prompt p and its corresponding rubrics, we first generate an image x using the current policy model. We then use the same fixed evaluator V (GPT-4.1-mini) as in Equation 3 to score each rubric

with the following system prompt. These binary scores are aggregated into the scalar training reward $r(x, p)$ used in Equation 1.

Prompt for RL Reward Generation

System_prompt:
You are evaluating whether an AI-generated image satisfies the following visual quality criterion.
You will receive a user prompt that used to generate the image and a visual quality evaluation criterion.
Your job is to return **1** if the image fully satisfies the criterion, or **0** if it clearly fails. Do not explain or elaborate.
Only output: 1 or 0. Do not include explanations, just return the value.

Text_prompt:
The prompt of the generated image that needs to be evaluated is: {user_prompt}
Evaluation Criterion ({key}):{criterion}
Scoring Instructions:
- Return 1 if the image fully satisfies the criterion.
- Return 0 if the image clearly fails.
Output: 1 or 0 only.

E.4 Prompt for the RL Data Generation Agent

Prompt for the RL Data Generation Agent

Below is a set of evaluation prompts. Each was generated directly into an image by that text-to-image model and scored by a rubric (higher reward = the model handles it well; lower reward = the model handles it poorly):
{testing_report}

Your task is to analyze the prompts across the full reward spectrum, with primary emphasis on the LOW-reward prompts. Identify the shared surface-level and visual-task characteristics of the kinds of prompts the model struggles with. Do NOT focus primarily on semantic topic similarity. Instead, look for recurring patterns such as:
- specific object types the model often fails on
- difficult attributes (e.g. color, count, size, shape, texture, material, age, pose, expression)
- difficult relations (e.g. left/right, above/below, in front of/behind, overlapping, holding, wearing)
- text-rendering demands (e.g. OCR text, signage, labels, logos, clothing text)
- visual style or medium (e.g. photorealistic, watercolor, vector art, 3D render, sketch, low-poly, cinematic, flat design)

Use both the average reward and the per-criterion failures to infer exactly what kinds of visual generation demands the model handles poorly.

Then generate {N_train} new training prompts with a mixed difficulty distribution:
- A clear majority should target the model's weak or recurring failure modes.
- A smaller portion should target medium-strength capabilities that still show some instability.
- A limited minority should target capabilities the model already handles relatively well, to preserve breadth and avoid overfitting only to the weakest areas.

Important:
- Do not generate prompts only from the single weakest category.
- Keep the distribution weighted toward weak capabilities, but include some prompts from stronger areas as well.
- The prompts should collectively improve the model in a balanced way while still prioritizing its main weaknesses.

Requirements for the prompts:
- They should target currently weak capabilities, especially recurring failure modes.
- Do not merely copy existing prompts; create new prompts that are diverse.
- Keep prompts natural and realistic for text-to-image model training.
- Make the prompts visually specific and testable.

Output format:
- Output exactly {N_train} lines.
- Each line must be valid JSONL: one JSON object per line.
- Do not output markdown fences.
- Do not output explanations, analysis, headings, bullets, or any text before or after the JSONL.
- Each JSON object must contain:
– ”id”: integer from 1 to {N_train}
– ”prompt”: the training prompt string

Example format:
{”id”:1,”prompt”:”...”}
{”id”:2,”prompt”:”...”}

Return only the JSONL prompt objects.

E.5 Prompt for the RL Data Generation Agent in the Reduced Diagnostic Criteria Ablation

Prompt for the RL Data Generation Agent in the *Reduced Diagnostic Criteria* Ablation

Below is a set of evaluation prompts. Each was generated directly into an image by that text-to-image model and scored by a rubric (higher reward = the model handles it well; lower reward = the model handles it poorly):
{testing_report}

Your task is to analyze the prompts across the full reward spectrum, with primary emphasis on the LOW-reward prompts. Identify the shared surface-level and visual-task characteristics of the kinds of prompts the model struggles with. Do NOT focus primarily on semantic topic similarity. Instead, look for recurring patterns such as:
- specific object types the model often fails on
- visual style or medium (e.g. photorealistic, watercolor, vector art, 3D render, sketch, low-poly, cinematic, flat design)

Use both the average reward and the per-criterion failures to infer exactly what kinds of visual generation demands the model handles poorly.

Then generate {N_train} new training prompts with a mixed difficulty distribution:
- A clear majority should target the model’s weak or recurring failure modes.
- A smaller portion should target medium-strength capabilities that still show some instability.
- A limited minority should target capabilities the model already handles relatively well, to preserve breadth and avoid overfitting only to the weakest areas.

Important:
- Do not generate prompts only from the single weakest category.
- Keep the distribution weighted toward weak capabilities, but include some prompts from stronger areas as well.
- The prompts should collectively improve the model in a balanced way while still prioritizing its main weaknesses.

Requirements for the prompts:
- They should target currently weak capabilities, especially recurring failure modes.
- Do not merely copy existing prompts; create new prompts that are diverse.
- Keep prompts natural and realistic for text-to-image model training.
- Make the prompts visually specific and testable.

Output format:
- Output exactly {N_train} lines.
- Each line must be valid JSONL: one JSON object per line.
- Do not output markdown fences.
- Do not output explanations, analysis, headings, bullets, or any text before or after the JSONL.
- Each JSON object must contain:
– ”id”: integer from 1 to {N_train}

– ”prompt”: the training prompt string

Example format:
{”id”:1,”prompt”:”...”}
{”id”:2,”prompt”:”...”}

Return only the JSONL prompt objects.

E.6 Prompt for the RL Data Generation Agent in the Hard-Only Sampling Ablation

Prompt for the RL Data Generation Agent in the *Hard-Only Sampling* Ablation

Below is a set of evaluation prompts. Each was generated directly into an image by that text-to-image model and scored by a rubric (higher reward = the model handles it well; lower reward = the model handles it poorly):
{testing_report}

Your task is to analyze the prompts across the full reward spectrum, with primary emphasis on the LOW-reward prompts. Identify the shared surface-level and visual-task characteristics of the kinds of prompts the model struggles with. Do NOT focus primarily on semantic topic similarity. Instead, look for recurring patterns such as:
- specific object types the model often fails on
- difficult attributes (e.g. color, count, size, shape, texture, material, age, pose, expression)
- difficult relations (e.g. left/right, above/below, in front of/behind, overlapping, holding, wearing)
- text-rendering demands (e.g. OCR text, signage, labels, logos, clothing text)
- visual style or medium (e.g. photorealistic, watercolor, vector art, 3D render, sketch, low-poly, cinematic, flat design)

Use both the average reward and the per-criterion failures to infer exactly what kinds of visual generation demands the model handles poorly.

Then generate {N_train} new training prompts that exclusively target the model’s weak or recurring failure modes. Prioritize the most frequent, severe, and persistent errors observed in the evaluation results, and avoid generating prompts that primarily assess capabilities the model already performs well on.

Requirements for the prompts:
- They should target currently weak capabilities, especially recurring failure modes.
- Do not merely copy existing prompts; create new prompts that are diverse.
- Keep prompts natural and realistic for text-to-image model training.
- Make the prompts visually specific and testable.

Output format:
- Output exactly {N_train} lines.
- Each line must be valid JSONL: one JSON object per line.
- Do not output markdown fences.
- Do not output explanations, analysis, headings, bullets, or any text before or after the JSONL.
- Each JSON object must contain:
– ”id”: integer from 1 to {N_train}
– ”prompt”: the training prompt string

Example format:
{”id”:1,”prompt”:”...”}
{”id”:2,”prompt”:”...”}

Return only the JSONL prompt objects.