Meta-TTRL: A Metacognitive Framework for Self-Improving Test-Time Reinforcement Learning for T2I Generation in Unified Multimodal Models

Lit Sin Tan[1,2*†] **Junzhe Chen**[1*] **Xiaolong Fu**[2] **Lichen Ma**[2]
Junshi Huang[2] **Jianzhong Shi**[2] **Yan Li**[2] **Lijie Wen**[1✉]

[1] School of Software, Tsinghua University [2] JD.com

{chen-lx26,chenjz24}@mails.tsinghua.edu.cn
wenlj@tsinghua.edu.cn

[*]Equal Contribution [†]Work done during an internship at JD.com [✉]Corresponding author

Abstract

Test-time scaling (TTS) improves text-to-image (T2I) generation in unified multimodal models (UMMs), but existing TTS methods typically rely on frozen-parameter search or sampling, yielding only ephemeral, instance-level improvements. We propose **Meta-TTRL**, a metacognitive test-time reinforcement learning (TTRL) framework that converts test-time generation experience into learning signals, enabling UMMs to self-improve during T2I generation without external reward models. **Meta-TTRL** formulates test-time learning as a closed-loop interaction between an object-level generator and a meta-level introspector. The introspector decomposes prompts into structured verification rubrics and produces confidence-enhanced intrinsic monitoring signals, which are used to optimize the generator policy through reinforcement learning (RL). Extensive experiments demonstrate that **Meta-TTRL** generalizes well across three representative UMMs, including Janus-Pro-7B, BAGEL, and Qwen-Image, achieving significant gains on compositional reasoning tasks and multiple T2I benchmarks with limited data. Further analyses with external introspectors, RL leakage, and alternative monitoring signals reveal a key principle for effective TTRL: metacognitive synergy, where monitoring signals must align with the model's own optimization regime.

1 Introduction

Unified multimodal models (UMMs) have emerged as a powerful paradigm for text-to-image (T2I) generation by integrating multimodal understanding and generation within a single architecture, improving semantic grounding, visual synthesis, image quality, and compositional fidelity [53, 8, 10, 51, 47, 9, 57]. Following the scaling trend of large pre-trained language models (LLMs) [6, 1], recent UMMs have adopted similar scaling strategies [8, 9], but their high training and deployment costs limit practical scalability and accessibility.

To reduce the cost of training-time scaling, recent studies explore test-time scaling (TTS), which improves T2I generation through additional test-time computation, such as candidate sampling and selection, self-verification, or iterative refinement [44, 15, 28, 46]. TTS demonstrate that UMMs can benefit from additional test-time computation without scaling model size and training data. However, existing TTS methods for T2I generation in UMMs improve generation performance primarily through test-time sampling or search strategies while keeping the model parameters fixed. As a result, the additional computation yields only instance-level improvements, leaving the UMM unable to

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

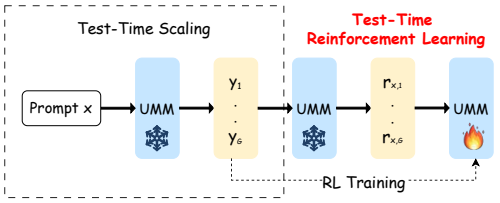


Figure 1: TTRL for T2I Generation in UMMs.

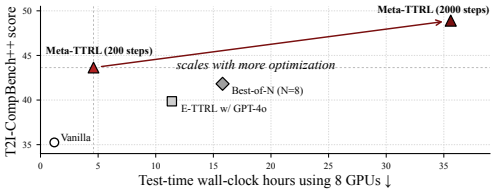


Figure 2: Performance-efficiency trade-off.

accumulate knowledge from prior inference experience. In other words, any performance gain is ephemeral: when encountering similar prompts in subsequent queries, the UMM often exhibits the same failure modes and must repeat the entire TTS procedure again. This observation naturally raises a key question: **Can the experience obtained during test-time scaling be leveraged for learning?**

A promising direction is test-time reinforcement learning (TTRL), which leverages test-time experience as learning signals to update model parameters at test time. TTRL has recently shown strong potential in language reasoning tasks [62] and vision–language understanding tasks [41]. However, despite its success in these tasks, TTRL for T2I generation in UMMs remains largely unexplored. This leads to a key question: *Can reinforcement learning enable effective and robust test-time learning for T2I generation in UMMs without relying on external reward models?*

To answer this question, we propose **Meta-TTRL**, a metacognitive test-time reinforcement learning framework for T2I generation in UMMs. As illustrated in Figure 1, **Meta-TTRL** extends TTS by leveraging the experience obtained during TTS as learning signals to update model parameters via reinforcement learning (RL). Following the Nelson–Narens model of metacognition [35], **Meta-TTRL** introduces a two-level metacognitive architecture in which the UMM operates at two levels: an object-level generator produces candidate images, while a meta-level introspector constructs structured evaluation rubrics and monitors generation outputs. The model-intrinsic monitoring signals, eliminate the need for external reward models while providing robust supervision and enabling self-improvement. Through the interaction between meta-level monitoring and object-level policy optimization, **Meta-TTRL** forms a monitoring–control loop that enables capability-level improvement, allowing performance gains to persist when the model encounters similar prompts.

Beyond proposing a new framework, we conduct a comprehensive analysis to better understand the potential of TTRL for T2I generation in UMMs in three aspects: (1) using external introspectors (E-TTRL) to assess whether stronger introspector improve TTRL; (2) RL leakage, where RL is applied directly to test data to estimate an upper bound; and (3) alternative monitoring signals, using a VQA model for evaluation instead of the introspector. Across these dimensions, we can better understand how **Meta-TTRL** leverages model-intrinsic introspection and achieves metacognitive synergy. Figure 2 highlights **Meta-TTRL**'s practical advantage: even with only 200 test-time optimization steps, it outperforms E-TTRL instantiated with GPT-4o [22] as the external introspector and Best-of-N ($N = 8$) while maintaining lower computational cost, and further gains are achieved with additional optimization. This underscores **Meta-TTRL**'s capability to achieve consistent and resource-efficient improvements in T2I generation.

Our contributions are summarized as follows:

- We introduce **Meta-TTRL**, the first TTRL framework for T2I generation in UMMs. **Meta-TTRL** leverages model-intrinsic introspection to construct structured verification rubrics and derive confidence-enhanced intrinsic monitoring signals for test-time policy optimization.
- We conduct controlled analyses to study the potential of TTRL from three aspects: external introspector (E-TTRL), RL leakage, and alternative monitoring signals. These analyses identify metacognitive synergy as a key factor for effective TTRL: monitoring signals should be aligned with the model's optimization regime.
- Extensive experiments show that **Meta-TTRL** consistently improves T2I generation across Janus-Pro-7B, BAGEL, and Qwen-Image. **Meta-TTRL** yields significant improvements in compositional reasoning tasks (e.g., shape, texture, and spatial reasoning), with relative gains exceeding 50% for weaker base models while also providing consistent benefits for stronger models. The improvements further generalize well to out-of-distribution settings, including cross-benchmark evaluations, indicating that **Meta-TTRL** does not overfit to specific benchmarks.

2 Related Work

Post-Training Reinforcement Learning for T2I in UMMs. Recent post-training RL methods for T2I generation in UMMs can be categorized by reward supervision. One line of work relies on external rewards, including human-preference reward models [52, 55, 25], unified multimodal reward models [49, 48], image–text alignment scores [40, 16, 29], MLLM-based reward models [19, 26, 61], and vision-language detectors [45, 30]. These rewards are commonly used in reinforcement learning from human feedback (RLHF) or reinforcement learning with verifiable rewards (RLVR) settings for UMM-based T2I generation [23, 31, 32, 59, 42, 37, 24], but their dependence on fixed external reward models may limit adaptability to diverse test-time prompts. Another line of work uses self-supervised or model-intrinsic rewards that exploit the coupling between multimodal understanding and generation, avoiding human annotations or external reward models [33, 17, 56, 36]. However, these methods are typically conducted as offline post-training before deployment. In contrast, our work treats T2I generation at test time as a learning process, enabling UMMs to update from test-time experience through model-intrinsic monitoring signals without relying on external reward models.

Test-Time Reinforcement Learning (TTRL). TTRL generalizes test-time training by leveraging RL with weak proxy rewards [62, 41]. In language reasoning, TTRL with majority voting can approach the performance of models trained on labeled test data [62]. Despite these advances, applying TTRL to T2I generation remains largely underexplored. In this work, we introduce **Meta-TTRL**, a metacognitive TTRL framework designed for T2I generation in UMMs.

3 Methodology

In this section, we present **Meta-TTRL**. At test time, the UMM performs intrinsic monitoring through introspection signals, assessing its generated outputs based on internal multimodal reasoning and understanding. These monitoring signals are then leveraged to guide test-time policy optimization, enabling self-improvement. Following the Nelson–Narens model of metacognition [35], **Meta-TTRL** is formulated as a metacognitive framework built upon a two-level architecture.

3.1 Two-level Metacognitive Architecture

Meta-TTRL implements a two-level architecture consisting of an **object level** and a **meta level**. The **object level** performs the core text-to-image generation task, while the **meta level** monitors the outputs of the object level and produces intrinsic monitoring signals. Formally, let $\mathcal{X}$ denote the space of prompts and $\mathcal{Y}$ the space of images. Given a test prompt $x \in \mathcal{X}$, the UMM operates as follows. At the object level, the UMM functions as a *generator* parameterized by a policy $\pi_\phi(\cdot \mid x)$, which produces candidate images $y \in \mathcal{Y}$. At the meta level, the UMM functions as an *introspector* parameterized by θ, which evaluates prompt–image pairs (x, y) and produces intrinsic monitoring signals used for policy optimization. This separation provides the structural basis for the monitoring–control loop described next.

3.2 Monitoring–Control Loop

Building on the two-level architecture, **Meta-TTRL** operates through a closed-loop interaction between monitoring and control. At each iteration, the object-level *generator* samples candidate images $y \sim \pi_\phi(\cdot \mid x)$. The meta-level *introspector* then evaluates each prompt–image pair (x, y) and maps it to a structured evaluation space, producing an intrinsic reward $r(x, y)$, which serves as the monitoring signal. Based on this monitoring signal, the meta level *introspector* controls the *generator* by adjusting the *generator* policy to maximize the reward:

$$\max_{\phi} \ \mathbb{E}_{y\sim\pi_\phi(\cdot|x)}[r(x, y)]. \tag{1}$$

In practice, the *generator* policy is updated via gradient-based optimization:

$$\phi \leftarrow \phi + \eta \nabla_\phi \mathbb{E}_{y\sim\pi_\phi(\cdot|x)}[r(x, y)], \tag{2}$$

where η denotes the learning rate. This monitoring–control loop enables the UMM to iteratively refine its generation policy through intrinsic monitoring signalsat test time. Building on this formulation, we implement the monitoring–control loop as a four-step procedure illustrated in Figure 3.

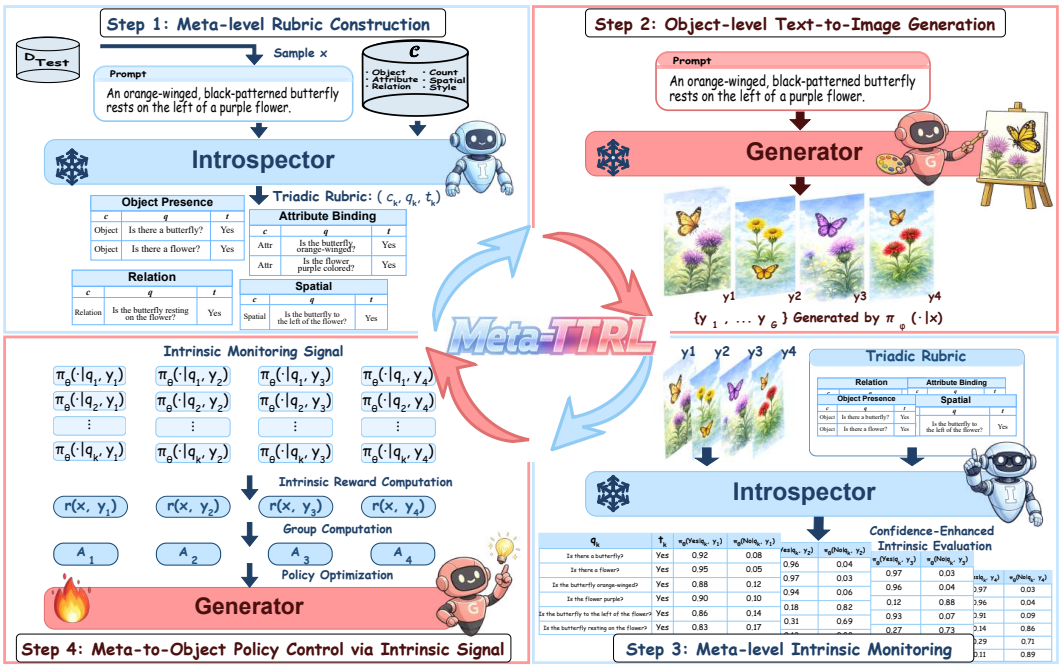


Figure 3: **Meta-TTRL** workflow: the meta-level *introspector* constructs rubrics and provides intrinsic monitoring signals, while the object-level *generator* produces candidate images. Signals are aggregated as rewards to guide policy optimization and improve T2I generation. For clarity, the figure omits question index m and shows representative verification questions.

3.3 Step 1: Meta-level Rubric Construction

Direct evaluation of complex prompts often leads to ambiguous or unstable reward signals [34, 2]. **Meta-TTRL** therefore introduces a rubric construction stage that decomposes prompts into structured evaluation criteria. Given the input prompt x, the meta-level *introspector* constructs an introspective rubric $\mathcal{Q}(x)$. The rubric $\mathcal{Q}(x)$ is constructed using a rubric schema that defines a set of cognitive dimensions: $\mathcal{C} = \{\text{Object}, \text{Attribute}, \text{Count}, \text{Spatial}, \text{Relation}, \text{Style}\}$. For each dimension $c_k \in \mathcal{C}$ that is applicable to the prompt x, the *introspector* derives a set of binary verification questions $q_{k,m}$ with corresponding target answers $t_{k,m} \in \{\text{Yes}, \text{No}\}$. This results in a structured rubric:

$$\mathcal{Q}(x) = \left\{ \left(c_k,\ \{q_{k,m},\ t_{k,m}\}_{m=1}^{M_k}\right) \right\}_{k=1}^{K}, \tag{3}$$

where M_k denotes the number of verification questions for dimension c_k. The rubric $\mathcal{Q}(x)$ defines a set of semantic constraints for intrinsic evaluation, which provide structured criteria for computing the intrinsic reward $r(x, y)$ in Eq. 1. Through rubric construction, the model's internal reasoning makes implicit generation intent explicit as evaluation criteria that serve as verifiable conditions for intrinsic monitoring (see Appendix B.1 for rubric construction details).

3.4 Step 2: Object-level Text-to-Image Generation

Given the prompt x, the object-level *generator* samples a group of candidate images $\{y_1, y_2, \ldots, y_G\} \sim \pi_\phi(\cdot \mid x)$, which represent diverse generations under the current policy and form the candidate outputs that will be evaluated using the rubric $\mathcal{Q}(x)$ constructed in Step 1.

3.5 Step 3: Meta-level Intrinsic Monitoring

Using the rubric $\mathcal{Q}(x)$ constructed in Step 1, the *introspector* evaluates each candidate image y_i generated in Step 2. For each candidate image y_i and each rubric criterion $(c_k, q_{k,m}, t_{k,m})$, the *introspector* estimates the confidence that the image satisfies the corresponding semantic constraint



Figure 4: Qualitative case studies of **Meta-TTRL**

$q_{k,m}$. Specifically, a confidence-enhanced score $s_{k,m}(y_i)$ is computed as:

$$s_{k,m}(y_i) = \frac{\pi_\theta(t_{k,m} \mid q_{k,m}, y_i)}{\sum_{t' \in \{\text{Yes,No}\}} \pi_\theta(t' \mid q_{k,m}, y_i)}, \tag{4}$$

where π_θ denotes the *introspector* policy. The intrinsic reward is then defined as the aggregated score across all rubric criteria:

$$r(x, y_i) = \exp\left(\frac{1}{\sum_{k=1}^{K} M_k} \sum_{k=1}^{K} \sum_{m=1}^{M_k} \log s_{k,m}(y_i)\right), \tag{5}$$

This reward is obtained by using the *introspector*'s introspection-based evaluation to evaluate how well the generated image satisfies the rubric criteria, thereby providing a monitoring signal for the subsequent meta-level control stage.

3.6 Step 4: Meta-to-Object Policy Control via Intrinsic Signal

Using the intrinsic reward $r(x, y_i)$ computed in Step 3, the *generator* policy π_ϕ is updated using meta-level intrinsic feedback to guide object-level generation. Direct optimization of absolute reward values can be unstable due to potential bias in self-evaluation. To improve robustness, we adopt Group Relative Policy Optimization (GRPO), which updates the policy based on the relative performance of multiple candidate generations. The policy update objective is defined as:

$$\mathcal{J}(\phi) = \frac{1}{G} \sum_{i=1}^{G} \left[\rho_i(\phi) A_i - \beta D_{\text{KL}}(\pi_\phi \| \pi_{\text{ref}})\right], \tag{6}$$

where the relative advantage A_i for image y_i is defined as:

$$A_i = \frac{r(x, y_i) - \text{mean}(\{r(x, y_j)\}_{j=1}^{G})}{\text{std}(\{r(x, y_j)\}_{j=1}^{G})}, \tag{7}$$

$\rho_i(\phi)$ denotes the importance sampling ratio between the current policy and the previous policy for sample y_i. D_{KL} is a KL divergence that regularizes the policy by constraining it to remain close to the reference policy π_{ref}. Intrinsic feedback derived from the *introspector*'s understanding and reasoning, grounded in introspection-based evaluation, is used to guide the *generator*'s policy updates. This allows introspection-based evaluation to shape object-level generation through rubric construction and intrinsic assessment, enabling test-time self-improvement without external supervision.

Table 1: Results on TIIF-Bench [50], T2I-CompBench++ [20], DPG-Bench [18].

Model	TIIF-Bench ↑		T2I-CompBench++ ↑								DPG ↑
	Short	Long	Color	Shape	Texture	2D Spatial	3D Spatial	Non-Spatial	Numeracy	Complex	Score
Qwen-Image	83.45	82.48	82.80	59.86	74.51	44.30	45.36	31.71	75.82	38.38	88.32
w/ Meta-TTRL	85.28	84.67	84.50	62.42	76.36	46.59	46.11	31.85	76.77	38.48	89.00
△	**+1.83**	**+2.19**	**+1.70**	**+2.56**	**+1.85**	**+2.29**	**+0.75**	**+0.14**	**+0.95**	**+0.10**	**+0.68**
	2.19%	2.66%	2.05%	4.28%	2.48%	5.17%	1.65%	0.44%	1.25%	0.26%	0.77%
BAGEL	71.65	70.32	80.81	57.94	71.54	34.03	37.78	31.11	65.52	37.78	84.03
w/ Meta-TTRL	75.98	74.59	82.69	61.94	74.55	38.08	43.69	31.28	67.00	38.79	86.33
△	**+4.33**	**+4.27**	**+1.88**	**+4.00**	**+3.01**	**+4.05**	**+5.91**	**+0.17**	**+1.48**	**+1.01**	**+2.30**
	6.04%	6.07%	2.33%	6.90%	4.21%	11.90%	15.64%	0.55%	2.26%	2.67%	2.74%
Janus-Pro-7B	64.41	66.94	52.02	32.72	40.39	15.73	27.37	31.27	44.50	37.98	83.84
w/ Meta-TTRL	71.42	70.74	79.33	50.10	67.52	32.46	37.51	31.35	54.08	38.88	85.67
△	**+7.01**	**+3.80**	**+27.31**	**+17.38**	**+27.13**	**+16.73**	**+10.14**	**+0.08**	**+9.58**	**+0.90**	**+1.83**
	10.88%	5.68%	52.50%	53.12%	67.17%	106.36%	37.04%	0.26%	21.53%	2.37%	2.18%

4 Experiments

4.1 Experimental Setup

Models. To evaluate the generalizability of **Meta-TTRL** across different architectural paradigms and model scales, we conduct experiments on three representative UMMs: Janus-Pro-7B [8], BAGEL [10], and Qwen-Image [51]. Our objective is to demonstrate that **Meta-TTRL** consistently improves T2I performance across diverse UMM architectures, including costly post-trained UMMs.

Benchmarks. We evaluate **Meta-TTRL** on three challenging T2I benchmarks: TIIF-Bench [50], T2I-CompBench++ [20], and DPG-Bench [18].

Baselines. Since TTRL has not been explored for T2I generation, we primarily compare **Meta-TTRL** with the corresponding base models to evaluate whether it can achieve effective improvements through self-improvement. Additional comparisons with other models are provided in Appendix D.

4.2 Main Results

Table 1 reports the results of **Meta-TTRL** on three challenging T2I benchmarks. In addition, Figure 4 shows qualitative case studies highlighting improvements in compositional fidelity and spatial reasoning. From these comparisons, we derive the following key conclusions:

(1) Meta-TTRL consistently improves performance across all evaluated models, including Qwen-Image, BAGEL, and Janus-Pro-7B. Notably, improvements are observed not only for weaker baselines but also for costly post-trained UMMs like Qwen-Image and BAGEL. Specifically, **Meta-TTRL** achieves average gains of 2.01, 4.30, and 5.41 for Qwen-Image, BAGEL, and Janus-Pro-7B, respectively, on the TIIF-Bench. **Meta-TTRL** results in significant total gains of 10.34, 22.55, and 109.25 on T2I-CompBench++ across eight compositional subdimensions for the three evaluated models.

(2) Meta-TTRL demonstrates pronounced improvements in attributes that require compositional reasoning, such as shape, texture, spatial reasoning, and numeracy. For instance, Qwen-Image shows improvements of 4.28% on shape and 5.17% on 2D spatial. BAGEL achieves gains of 6.90%, 11.90%, and 15.64% on shape, 2D spatial, and 3D spatial attributes, respectively. These improvements indicate that the introspector's internal evaluation provides useful and effective guidance during training, enabling UMMs to better capture attribute dependencies and relational structures in T2I generation.

(3) We observe that **Meta-TTRL** leads to more dramatic relative improvements for models with weaker compositional reasoning at baseline. For example, Janus-Pro-7B, which lags behind Qwen-Image and BAGEL on T2ICompBench++, benefits from significant improvements such as 52.50%

on color, 53.12% on shape, 67.17% on texture, and 106.36% on 2D spatial. In comparison, stronger models like Qwen-Image and BAGEL show moderate but consistently positive improvements. This can be explained by the fact that the monitoring signals in **Meta-TTRL** provide fine-grained guidance during test-time optimization, which is particularly helpful for models with weaker baseline generative performance, allowing them to better align generated outputs with the rubric criteria.

(4) Beyond benchmark-specific improvements, **Meta-TTRL** generalizes well to out-of-distribution settings. We first evaluate its generalization on the GenEval benchmark [13]. As shown in Table 2, **Meta-TTRL** consistently improves performance across all three evaluated models. Further, we evaluate cross-benchmark generalization using Janus-Pro-7B. As shown in Table 3, we apply **Meta-TTRL** on each T2I benchmark and evaluating T2I performance on the others. Despite substantial differences in prompt distributions, evaluation metrics, and judging protocols, **Meta-TTRL** maintains consistent improvements across all settings, indicating that **Meta-TTRL** does not overfit to specific benchmarks and acquires generalizable gains in image generation quality.

Table 2: Performance on GenEval for all evaluated models

Train on / Model	T2I-CompBench++		TIIF-Bench		DPG-Bench	
	Vanilla	**w/ Meta-TTRL**	Vanilla	**w/ Meta-TTRL**	Vanilla	**w/ Meta-TTRL**
Janus-Pro-7B	77.80	81.14 **+3.34**	77.80	79.40 **+1.60**	77.80	78.45 **+0.65**
Bagel	79.24	83.37 **+4.13**	79.24	82.67 **+3.43**	79.24	82.67 **+3.43**
Qwen-Image	87.00	88.52 **+1.52**	87.00	89.72 **+2.72**	87.00	88.43 **+1.43**

Table 3: Cross-benchmark generalization on Janus-Pro-7B.

Eval on / Train on	T2I-CompBench++		TIIF-Bench		DPG-Bench	
	Vanilla	**w/ Meta-TTRL**	Vanilla	**w/ Meta-TTRL**	Vanilla	**w/ Meta-TTRL**
T2I-CompBench++	35.25	48.90 **+13.65**	65.68	68.71 **+3.03**	83.84	84.63 **+0.79**
TIIF-Bench	35.25	40.44 **+5.19**	65.68	71.08 **+5.40**	83.84	84.32 **+0.48**
DPG-Bench	35.25	38.93 **+3.68**	65.68	66.71 **+1.03**	83.84	85.67 **+1.83**

5 Analysis

In this section, we empirically investigate the potential of TTRL for improving T2I generation in UMMs through controlled analyses. Specifically, we compare **Meta-TTRL** with three variants:

(1) External Introspector (E-TTRL): the introspector is replaced by external models, including the closed-source GPT-4o and the open-source Qwen3-VL-235B-A22B-Instruct [3], to study the effect of introspector capability on TTRL.

(2) RL Leakage: RL is performed directly on test data using UnifiedReward [49], a widely used reward model for image generation, providing an approximate empirical upper bound for TTRL.

(3) Alternative Monitoring Signals: rubric construction is retained with the introspector while rubric evaluation is performed by the visual question answering (VQA) model, GIT [45], to examine whether alternative monitoring mechanisms can support effective TTRL.

For fair comparison, we fix the number of RL steps for each model on each benchmark across all experiments (see Appendix B.1 for details).

5.1 Test-Time Reinforcement Learning with External Introspectors (E-TTRL)

Does a stronger external introspector lead to further improvements in TTRL? A natural hypothesis is that leveraging the stronger reasoning and multimodal understanding of an external model may provide more informative signals and thus lead to more effective TTRL. To test this hypothesis, we study **E-TTRL**, where the introspector in **Meta-TTRL** is replaced with an external model, with the monitoring–control pipeline remains unchanged, and both rubric construction and rubric evaluation are performed by the external introspector. We consider two configurations: **(1)** a strong closed-source multimodal judge, GPT-4o; **(2)** a large-scale open-source multimodal model,

Qwen3-VL-235B-A22B-Instruct. This experiment allows us to isolate whether improvements come from stronger capability or intrinsic alignment between the generator and introspective signals.

E-TTRL with closed-source GPT-4o as the external introspector.

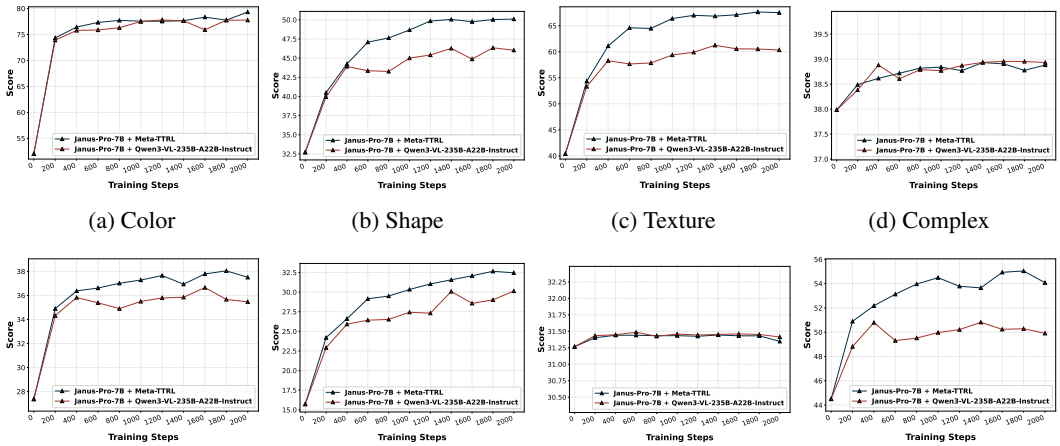
We compare **Meta-TTRL** with E-TTRL instantiated using GPT-4o under the same 200-step optimization budget due to API cost and latency, with results reported in Table 4. From this comparison, we derive the following findings and analysis: **(1) Meta-TTRL** consistently outperforms GPT-4o-based E-TTRL in overall performance. Althougb GPT-4o improves the overall score from 35.25 to 39.86, **Meta-TTRL** further increases it to 43.64 and achieves greater gains on compositional attributes such as 3D spatial reasoning, color, 2D spatial reasoning, shape, and texture. **(2)** The performance gap is driven by more effective credit assignment. GPT-4o provides coarse, discrete feedback (e.g., yes/no), which is insufficient to guide fine-grained compositional optimization, while **Meta-TTRL** leverages confidence-enhanced introspective rewards to produce more informative and optimization-aligned signals. **(3)** GPT-4o incurs substantially higher computational overhead despite weaker performance. E-TTRL with GPT-4o exhibits a reward latency of 46.93 seconds and requires 10.2 hours for 200 optimization steps, compared to 11.16 seconds and 3.4 hours for **Meta-TTRL**, making **Meta-TTRL** both more effective and more efficient in practice.

Table 4: Results of **Meta-TTRL** and **E-TTRL with GPT-4o** on T2I-CompBench++

Janus-Pro-7B	Color	Shape	Texture	2D Spatial	3D Spatial	Non-Spatial	Numeracy	Complex	Overall
Vanilla	52.02	32.72	40.39	15.73	27.37	31.27	44.50	37.98	35.25
w/ E-TTRL GPT-4o	66.01	36.59	46.70	21.13	31.54	**31.41**	47.27	38.26	39.86
w/ Meta-TTRL	**74.35**	**40.50**	**54.36**	**24.19**	**34.90**	31.40	**50.89**	**38.49**	**43.64**

E-TTRL with open-source Qwen3-VL-235B-A22B-Instruct as the external introspector.

We compare **Meta-TTRL** with E-TTRL instantiated using Qwen3-VL-235B-A22B-Instruct. Figure 5 shows the performance across training steps on eight subdimensions. From this comparison, we derive the following findings and analysis: **(1) Meta-TTRL** consistently outperforms Qwen3-VL-235B-based E-TTRL on six out of eight subdimension. In particular, **Meta-TTRL** achieves substantial improvements in shape, texture, and numeracy, while maintaining comparable performance on the remaining dimensions. **(2)** The performance gap is attributed to optimization misalignment. Although Qwen3-VL-235B-A22B-Instruct has stronger reasoning capability, its monitoring signals often lie beyond the effective optimization regime of the base model. This misalignment limits its utility for gradient-based test-time optimization. In contrast, intrinsic introspection in **Meta-TTRL** aligns more effectively with the base model's optimization regime, where this metacognitive synergy facilitates a more stable and efficient learning process and induces curriculum-like test-time learning.



(a) Color (b) Shape (c) Texture (d) Complex

(e) 3D Spatial (f) 2D Spatial (g) Non-spatial (h) Numeracy

Figure 5: Performance comparison between **Meta-TTRL (blue)** and **E-TTRL with Qwen3-VL-235B-A22B-Instruct (red)** on T2I-CompBench++ across eight subdimensions.

E-TTRL with comparable-scale Qwen2.5-VL-7B-Instruct as the external introspector.

To rule out the effect of optimization noise from an over-scaled external teacher, we further compare **Meta-TTRL** with E-TTRL using Qwen2.5-VL-7B-Instruct [4], whose scale is comparable to Janus-Pro-7B. As reported in Table 5, Qwen2.5-VL-7B obtains 45.08 overall, close to the 46.25 achieved by the much larger Qwen3-VL-235B-A22B-Instruct. In contrast, **Meta-TTRL** achieves 48.90, outperforming both external introspectors, especially on texture, shape, and numeracy.

Table 5: Comparison of **Meta-TTRL** and E-TTRL using Qwen introspectors on T2I-CompBench++

Janus-Pro-7B	Color	Shape	Texture	2D Spatial	3D Spatial	Non-Spatial	Numeracy	Complex	Overall
Vanilla	52.02	32.72	40.39	15.73	27.37	31.27	44.50	37.98	35.25
w/ E-TTRL Qwen2.5-VL-7B-Instruct	75.99	43.46	58.05	27.54	36.04	31.38	49.32	38.85	45.08
w/ E-TTRL Qwen3-VL-235B-A22B-Instruct	77.76	46.04	60.35	30.14	35.47	**31.41**	49.91	38.93	46.25
w/ Meta-TTRL	**79.33**	**50.10**	**67.52**	**32.46**	**37.51**	31.35	**54.08**	**38.88**	**48.90**

Insight: The E-TTRL results show that simply using a stronger or larger external evaluator does not guarantee better TTRL. Instead, the effectiveness of TTRL depends less on the evaluator's absolute capability and more on whether the monitoring signals are capacity-matched, highlighting the critical role of metacognitive synergy in enhancing TTRL performance.

5.2 RL Leakage

How much improvement can RL leakage bring to TTRL? To further explore the potential of TTRL, we consider RL leakage, where RL is directly applied to test data using UnifiedReward as the reward model. Although this violates the standard train–test separation in post-training RL, it allows us to study how much improvement RL leakage can bring to TTRL. We conduct experiments with Janus-Pro-7B, Bagel, and Qwen-Image on three T2I benchmarks, with results shown in Figure 6 and Figure 7. Overall, **Meta-TTRL** consistently outperforms RL Leakage on TIIF-Bench and DPG-Bench, and achieves better performance on most T2I-CompBench++ subdimensions across all evaluated models. On the remaining dimensions, **Meta-TTRL** and RL Leakage perform comparably. These results suggest that, through metacognitive synergy, **Meta-TTRL** produces more effective monitoring signals than a specially trained reward model, enabling stronger test-time self-improvement.

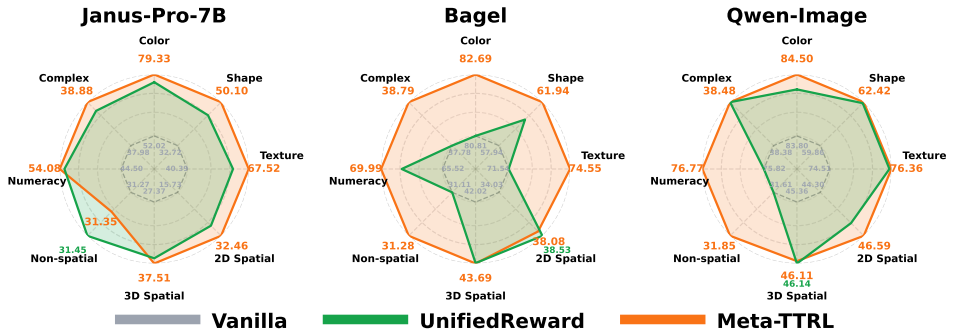


Figure 6: Performance comparison on T2I-CompBench++ subdimensions.

5.3 Alternative Monitoring Signals

Can alternative monitoring signals improve TTRL? To further examine the role of monitoring signals, we replace the rubric evaluation component in **Meta-TTRL** with a VQA model while keeping introspector-based rubric construction unchanged. We use GIT-large-VQAv2 [14], since VQA models are widely used as reward models for T2I reinforcement learning [23, 38]. We evaluate Janus-Pro-7B, Bagel, and Qwen-Image on three T2I benchmarks, with results shown in Table 6. Overall, GIT-based rubric evaluation improves the base models but consistently underperforms **Meta-TTRL**. We attribute

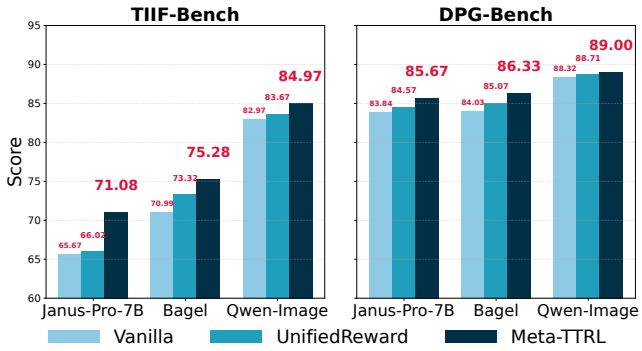


Figure 7: Performance comparison on TIIF-Bench and DPG-Bench.

this gap to the inconsistency between rubric construction and evaluation: rubrics are generated by the introspector but assessed by an external VQA model, which may not fully capture their intended criteria. This mismatch introduces noisy monitoring signals, weakens the metacognitive synergy between monitoring and learning, and ultimately reduces the effectiveness of TTRL.

Table 6: Comparison of **Meta-TTRL** and VQA-based monitoring signals on T2I benchmarks.

Train on	Janus-Pro-7B			Bagel			Qwen-Image		
	Vanilla	GIT	Meta-TTRL	Vanilla	GIT	Meta-TTRL	Vanilla	GIT	Meta-TTRL
T2I++	35.25	47.16 +11.91	48.90 **+13.65**	52.60	54.83 +2.23	55.13 **+2.53**	56.71	57.71 +1.00	57.88 **+1.17**
TIIF	65.68	69.73 +4.05	71.08 **+5.40**	71.00	72.70 +1.70	75.29 **+4.29**	82.97	84.24 +1.27	84.98 **+2.01**
DPG	83.84	84.78 +0.94	85.67 **+1.83**	84.03	86.24 +2.21	86.33 **+2.30**	88.32	88.83 +0.51	89.00**+0.68**

6 Conclusion and Limitations

Conclusion. We introduce **Meta-TTRL**, a metacognitive TTRL framework for T2I generation in UMMs. By using model-intrinsic introspection to construct structured rubrics and provide confidence-enhanced monitoring signals, Meta-TTRL enables test-time self-improvement without external reward models. Experiments across Janus-Pro-7B, BAGEL, and Qwen-Image show consistent gains on multiple T2I benchmarks, especially on compositional reasoning dimensions. Our analyses further show that effective TTRL depends not only on reward quality, but also on the alignment between monitoring signals and the model's optimization regime, which we identify as metacognitive synergy.
Limitations. Meta-TTRL requires access to the model's parameters, which prevents application to closed-source models.

7 Acknowledgements and Disclosures

This work is supported by the National Key Research and Development Program of China (No.2024YFB3309702), the National Nature Science Foundation of China (No.62021002), Tsinghua BNRist and Beijing Key Laboratory of Industrial Big Data System and Application.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] Ying Ba, Tianyu Zhang, Yalong Bai, Wenyi Mo, Tao Liang, Bing Su, and Ji-Rong Wen. Enhancing reward models for high-quality image generation: Beyond text-image alignment. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19022–19031, 2025.

[3] 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.

[4] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report, 2025.

[5] James Betker, Gabriel Goh, Li Jing, Tim Brooks, Jianfeng Wang, Linjie Li, Long Ouyang, Juntang Zhuang, Joyce Lee, Yufei Guo, et al. Improving image generation with better captions. *Computer Science. https://cdn. openai. com/papers/dall-e-3. pdf*, 2(3):8, 2023.

[6] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[7] Junsong Chen, Chongjian Ge, Enze Xie, Yue Wu, Lewei Yao, Xiaozhe Ren, Zhongdao Wang, Ping Luo, Huchuan Lu, and Zhenguo Li. Pixart-σ: Weak-to-strong training of diffusion transformer for 4k text-to-image generation. In *European Conference on Computer Vision*, pages 74–91. Springer, 2024.

[8] 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, 2025.

[9] Yufeng Cui, Honghao Chen, Haoge Deng, Xu Huang, Xinghang Li, Jirong Liu, Yang Liu, Zhuoyan Luo, Jinsheng Wang, Wenxuan Wang, et al. Emu3. 5: Native multimodal models are world learners. *arXiv preprint arXiv:2510.26583*, 2025.

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

[11] Runpei Dong, Chunrui Han, Yuang Peng, Zekun Qi, Zheng Ge, Jinrong Yang, Liang Zhao, Jianjian Sun, Hongyu Zhou, Haoran Wei, et al. Dreamllm: Synergistic multimodal comprehension and creation. *arXiv preprint arXiv:2309.11499*, 2023.

[12] 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.

[13] 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.

[14] Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6904–6913, 2017.

[15] Ziyu Guo, Renrui Zhang, Chengzhuo Tong, Zhizheng Zhao, Rui Huang, Haoquan Zhang, Manyuan Zhang, Jiaming Liu, Shanghang Zhang, Peng Gao, et al. Can we generate images with cot? let's verify and reinforce image generation step by step. *arXiv preprint arXiv:2501.13926*, 2025.

[16] Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. Clipscore: A reference-free evaluation metric for image captioning. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, pages 7514–7528, 2021.

[17] Jixiang Hong, Yiran Zhang, Guanzhong Wang, Yi Liu, Ji-Rong Wen, and Rui Yan. Reinforcing multimodal understanding and generation with dual self-rewards. *arXiv e-prints*, pages arXiv–2506, 2025.

[18] Xiwei Hu, Rui Wang, Yixiao Fang, Bin Fu, Pei Cheng, and Gang Yu. Ella: Equip diffusion models with llm for enhanced semantic alignment, 2024.

[19] Yushi Hu, Benlin Liu, Jungo Kasai, Yizhong Wang, Mari Ostendorf, Ranjay Krishna, and Noah A Smith. Tifa: Accurate and interpretable text-to-image faithfulness evaluation with question answering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 20406–20417, 2023.

[20] Kaiyi Huang, Chengqi Duan, Kaiyue Sun, Enze Xie, Zhenguo Li, and Xihui Liu. T2i-compbench++: An enhanced and comprehensive benchmark for compositional text-to-image generation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.

[21] Kaiyi Huang, Kaiyue Sun, Enze Xie, Zhenguo Li, and Xihui Liu. T2i-compbench: A comprehensive benchmark for open-world compositional text-to-image generation. *Advances in Neural Information Processing Systems*, 36:78723–78747, 2023.

[22] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

[23] Dongzhi Jiang, Ziyu Guo, Renrui Zhang, Zhuofan Zong, Hao Li, Le Zhuo, Shilin Yan, Pheng-Ann Heng, and Hongsheng Li. T2i-r1: Reinforcing image generation with collaborative semantic-level and token-level cot. *arXiv preprint arXiv:2505.00703*, 2025.

[24] Jingjing Jiang, Chongjie Si, Jun Luo, Hanwang Zhang, and Chao Ma. Co-reinforcement learning for unified multimodal understanding and generation. *arXiv preprint arXiv:2505.17534*, 2025.

[25] Yuval Kirstain, Adam Polyak, Uriel Singer, Shahbuland Matiana, Joe Penna, and Omer Levy. Pick-a-pic: An open dataset of user preferences for text-to-image generation. *Advances in neural information processing systems*, 36:36652–36663, 2023.

[26] Max Ku, Dongfu Jiang, Cong Wei, Xiang Yue, and Wenhu Chen. Viescore: Towards explainable metrics for conditional image synthesis evaluation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12268–12290, 2024.

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

[28] Leon Liangyu Chen, Haoyu Ma, Zhipeng Fan, Ziqi Huang, Animesh Sinha, Xiaoliang Dai, Jialiang Wang, Zecheng He, Jianwei Yang, Chunyuan Li, et al. Unit: Unified multimodal chain-of-thought test-time scaling. *arXiv e-prints*, pages arXiv–2602, 2026.

[29] Zhiqiu Lin, Deepak Pathak, Baiqi Li, Jiayao Li, Xide Xia, Graham Neubig, Pengchuan Zhang, and Deva Ramanan. Evaluating text-to-visual generation with image-to-text generation. In *European Conference on Computer Vision*, pages 366–384. Springer, 2024.

[30] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. In *European conference on computer vision*, pages 38–55. Springer, 2024.

[31] Yifu Luo, Xinhao Hu, Keyu Fan, Haoyuan Sun, Zeyu Chen, Bo Xia, Tiantian Zhang, Yongzhe Chang, and Xueqian Wang. Reinforcement learning meets masked generative models: Mask-grpo for text-to-image generation. *arXiv preprint arXiv:2510.13418*, 2025.

[32] Xiaoxiao Ma, Haibo Qiu, Guohui Zhang, Zhixiong Zeng, Siqi Yang, Lin Ma, and Feng Zhao. Stage: Stable and generalizable grpo for autoregressive image generation. *arXiv preprint arXiv:2509.25027*, 2025.

[33] Weijia Mao, Zhenheng Yang, and Mike Zheng Shou. Unirl: Self-improving unified multimodal models via supervised and reinforcement learning. *arXiv preprint arXiv:2505.23380*, 2025.

[34] Ninareh Mehrabi, Palash Goyal, Apurv Verma, Jwala Dhamala, Varun Kumar, Qian Hu, Kai-Wei Chang, Richard Zemel, Aram Galstyan, and Rahul Gupta. Resolving ambiguities in text-to-image generative models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14367–14388, 2023.

[35] Thomas O Nelson. Metamemory: A theoretical framework and new findings. In *Psychology of learning and motivation*, volume 26, pages 125–173. Elsevier, 1990.

[36] Jiadong Pan, Liang Li, Yuxin Peng, Yu-Ming Tang, Shuohuan Wang, Yu Sun, Hua Wu, Qingming Huang, and Haifeng Wang. Learning to generate via understanding: Understanding-driven intrinsic rewarding for unified multimodal models. *arXiv preprint arXiv:2603.06043*, 2026.

[37] Kaihang Pan, Yang Wu, Wendong Bu, Kai Shen, Juncheng Li, Yingting Wang, Yunfei Li, Siliang Tang, Jun Xiao, Fei Wu, et al. Janus-pro-r1: Advancing collaborative visual comprehension and generation via reinforcement learning. *arXiv preprint arXiv:2506.01480*, 2025.

[38] Jiho Park, Sieun Choi, Jaeyoon Seo, and Jihie Kim. Stablesketcher: Enhancing diffusion model for pixel-based sketch generation via visual question answering feedback. *arXiv preprint arXiv:2510.20093*, 2025.

[39] 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. *arXiv preprint arXiv:2307.01952*, 2023.

[40] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

[41] Akshit Singh, Shyam Marjit, Wei Lin, Paul Gavrikov, Serena Yeung-Levy, Hilde Kuehne, Rogerio Feris, Sivan Doveh, James Glass, and M Jehanzeb Mirza. Ttrv: Test-time reinforcement learning for vision language models. *arXiv preprint arXiv:2510.06783*, 2025.

[42] Jiangtong Tan, Lin Liu, Jie Huanng, Xiaopeng Zhang, Qi Tian, and Feng Zhao. Parauni: Enhance generation in unified multimodal model with reinforcement-driven hierarchical parallel information interaction. *arXiv preprint arXiv:2512.05422*, 2025.

[43] Chameleon Team. Chameleon: Mixed-modal early-fusion foundation models, 2024. *URL https://arxiv. org/abs/2405.09818*, 9(8), 2024.

[44] Rui Tian, Mingfei Gao, Mingze Xu, Jiaming Hu, Jiasen Lu, Zuxuan Wu, Yinfei Yang, and Afshin Dehghan. Unigen: Enhanced training & test-time strategies for unified multimodal understanding and generation. *arXiv preprint arXiv:2505.14682*, 2025.

[45] Jianfeng Wang, Zhengyuan Yang, Xiaowei Hu, Linjie Li, Kevin Lin, Zhe Gan, Zicheng Liu, Ce Liu, and Lijuan Wang. Git: A generative image-to-text transformer for vision and language. *arXiv preprint arXiv:2205.14100*, 2022.

[46] 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.

[47] Xinlong Wang, Xiaosong Zhang, Zhengxiong Luo, Quan Sun, Yufeng Cui, Jinsheng Wang, Fan Zhang, Yueze Wang, Zhen Li, Qiying Yu, et al. Emu3: Next-token prediction is all you need. *arXiv preprint arXiv:2409.18869*, 2024.

[48] 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*, 2025.

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

[50] Xinyu Wei, Jinrui Zhang, Zeqing Wang, Hongyang Wei, Zhen Guo, and Lei Zhang. Tiif-bench: How does your t2i model follow your instructions? *arXiv preprint arXiv:2506.02161*, 2025.

[51] 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, 2025.

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

[53] Jinheng Xie, Weijia Mao, Zechen Bai, David Junhao Zhang, Weihao Wang, Kevin Qinghong Lin, Yuchao Gu, Zhijie Chen, Zhenheng Yang, and Mike Zheng Shou. Show-o: One single transformer to unify multimodal understanding and generation. *arXiv preprint arXiv:2408.12528*, 2024.

[54] Yi Xin, Siqi Luo, Qi Qin, Haoxing Chen, Kaiwen Zhu, Zhiwei Zhang, Yangfan He, Rongchao Zhang, Jinbin Bai, Shuo Cao, et al. dmllm-tts: Self-verified and efficient test-time scaling for diffusion multi-modal large language models. *arXiv preprint arXiv:2512.19433*, 2025.

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

[56] Zhiyuan Yan, Kaiqing Lin, Zongjian Li, Junyan Ye, Hui Han, Haochen Wang, Zhendong Wang, Bin Lin, Hao Li, Xinyan Xiao, Jingdong Wang, Haifeng Wang, and Li Yuan. Unified multimodal models as auto-encoders, 2026.

[57] Ling Yang, Ye Tian, Bowen Li, Xinchen Zhang, Ke Shen, Yunhai Tong, and Mengdi Wang. Mmada: Multimodal large diffusion language models. *arXiv preprint arXiv:2505.15809*, 2025.

[58] Haofei Yu, Zhengyang Qi, Lawrence Keunho Jang, Russ Salakhutdinov, Louis-Philippe Morency, and Paul Pu Liang. Mmoe: Enhancing multimodal models with mixtures of multimodal interaction experts. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 10006–10030, 2024.

[59] Guohui Zhang, Hu Yu, Xiaoxiao Ma, JingHao Zhang, Yaning Pan, Mingde Yao, Jie Xiao, Linjiang Huang, and Feng Zhao. Group critical-token policy optimization for autoregressive image generation. *arXiv preprint arXiv:2509.22485*, 2025.

[60] Xinchen Zhang, Xiaoying Zhang, Youbin Wu, Yanbin Cao, Renrui Zhang, Ruihang Chu, Ling Yang, and Yujiu Yang. Generative universal verifier as multimodal meta-reasoner. *arXiv preprint arXiv:2510.13804*, 2025.

[61] Shijie Zhou, Ruiyi Zhang, Huaisheng Zhu, Branislav Kveton, Yufan Zhou, Jiuxiang Gu, Jian Chen, and Changyou Chen. Multimodal llms as customized reward models for text-to-image generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19638–19648, 2025.

[62] Yuxin Zuo, Kaiyan Zhang, Li Sheng, Shang Qu, Ganqu Cui, Xuekai Zhu, Haozhan Li, Yuchen Zhang, Xinwei Long, Ermo Hua, et al. Ttrl: Test-time reinforcement learning. *arXiv preprint arXiv:2504.16084*, 2025.

A More Related Work

Unified Multimodal Models (UMMs). Unified multimodal models (UMMs) jointly support multimodal understanding and generation. Autoregressive methods extend the next-token prediction paradigm to jointly model text and discrete visual tokens [43, 47, 8]. Diffusion methods couple language models with diffusion modules for image generation [11, 51]. Unified transformer architectures integrate language modeling and diffusion within single architectures [58, 10]. Our framework generalizes across all three architectural paradigms and is evaluated on three representative UMMs, including Janus-Pro-7B [8], Qwen-Image [51], and BAGEL [10].

Test-Time Scaling (TTS) for T2I in UMMs. Test-time scaling (TTS) improves T2I generation in UMMs by using additional inference-time computation instead of further scaling model size and training data. Existing methods mainly follow two paradigms: parallel sampling-based scaling, which selects the best image from multiple candidates using external reward models, verifiers, or self-verification signals [15, 44]; and sequential iterative scaling, which progressively reasons, evaluates, and refines outputs across multiple inference rounds [28, 46, 54, 60]. However, these methods keep model parameters frozen, yielding instance-level gains without accumulating experience. In contrast, our work turns test-time experience into learning signals for persistent capability-level improvement.

B More Experiment Details

B.1 Rubric Construction Details

We implement the introspector using a constrained prompting template that enforces:

- strict grounding to the input text,
- binary (Yes/No) verifiability,
- a one-to-one correspondence between question forms and cognitive dimensions (each question is prepended with a dimension tag).

For benchmarks that provide prompt-aligned verification questions (e.g., DPG and TIIF , we directly adopt them as $Q(x)$. Otherwise, $Q(x)$ is generated using the constrained introspector template (see Fig. 8 for details).

For Janus-Pro, we use a more fine-grained version of the template that decomposes the process into two explicit steps (fact extraction $\rightarrow$ question conversion), which reduce hallucination and enforce deterministic output format (see Fig. 9 for details).

Implementation Details and Hyperparameters For unified-parameter UMMs (e.g., Janus-Pro), we freeze the introspector by setting $\theta \leftarrow \theta_0$ and keeping θ fixed throughout training, while optimizing the generator parameters ϕ at test time.

For UMMs with distinct generation and understanding modules (e.g., Bagel and Qwen-Image), only the generation-side ϕ is updated via LoRA injection, keeping the understanding-side parameters θ frozen. This stabilizes training by preventing intrinsic reward drift and mitigating reward hacking during test-time optimization.

All test-time hyperparameters, including those for GRPO, are fixed across tasks for each backbone model. Detailed configurations are summarized in Tables 7, 8, and 9.

INTROSPECTOR_SYSTEM_PROMPT_TEMPLATE

```
INTROSPECTOR_SYSTEM_PROMPT_TEMPLATE = """
You are a constrained generator.
Your task is to generate binary verification questions based strictly on the input prompt.
For each question, you must PREPEND the correct cognitive dimension tag.

### COGNITIVE DIMENSIONS (Select from this list)
- [Object]: Existence of entities.
- [Attribute]: Colors, shapes, states.
- [Count]: Specific quantities.
- [Spatial]: Position (left, right, on, under).
- [Relation]: Actions (holding, wearing).
- [Style]: Art style.

### STRICT RULES
1. **Fact-Based**: Only ask about what is EXPLICITLY in the text.
2. **Tagging**: Every question must start with a [Dimension] tag.
3. **FORBIDDEN**: "What", "How", "Where", "Which". (e.g., "What color is..." -> BANNED)
4. **MANDATORY**: Start every question with "Is" or "Are".
5. **Silence**: If a dimension is not in the text, DO NOT generate a question for it.
6. **Stop**: After listing the questions for the current prompt, output "###" and STOP.

### EXAMPLES

Input: "A blue bench and a red orange"
Output:
1. [Object] Is there a bench?
2. [Object] Is there a orange?
3. [Attribute] Is the bench blue?
4. [Attribute] Is the orange red?
###

Input: "The wallet lies near a horse."
Output:
1. [Object] Is there a wallet?
2. [Object] Is there a horse?
3. [Spatial] Is the wallet near the horse?
###

Input: "Five drums."
Output:
1. [Count] Are there exactly five drums in the image?
###

Input: "{PROMPT}"
Output:
1."""
```

Figure 8: Introspector System Prompt Template used for Bagel and Qwen-Image models

INTROSPECTOR_SYSTEM_PROMPT_TEMPLATE

```
INTROSPECTOR_SYSTEM_PROMPT_TEMPLATE = """
You are a constrained generator. You must NOT write open-ended questions.

Goal: Convert the input prompt into binary VQA questions.

You MUST follow the procedure below exactly.

========================
STEP A: Extract facts
========================
Extract ONLY explicit constraints from the prompt as facts.
Output facts as one per line using one of the following formats:

ENTITY: <entity>
ATTR: <entity> | <attribute>
COUNT: <number> | <entity_plural>
SPATIAL: <entity1> | <relation> | <entity2>
REL: <subject> | <action> | <object>
STYLE: <style>

Rules for STEP A:
- Do NOT invent anything.
- Use short noun phrases for entities.
- If a piece of information is not explicit, omit it.

========================
STEP B: Convert facts to questions
========================
Convert EACH fact into exactly ONE yes/no question using ONLY the templates below.
You MUST choose the template that matches the fact type.

TEMPLATES (you must copy them exactly, only replace <...>):
- [Object] Is there a/an <entity> in the image?
- [Attribute] Is the <entity> <attribute>?
- [Count] Are there exactly <number> <entity_plural> in the image?
- [Spatial] Is the <entity1> <relation> the <entity2>?
- [Relation] Is the <subject> <action> the <object>?
- [Style] Is the image in <style> style?

Hard rules for STEP B:
- Every question MUST start with "Is" or "Are".
- Do NOT use the words: What, How, Where, Which.
- Output ONLY the questions, numbered "1.", "2.", ...
- After the last question output a line with exactly: ###
- STOP immediately after ###

========================
EXAMPLES
========================

INPUT:
A blue bench and a red orange

FACTS:
ENTITY: bench
ENTITY: orange
ATTR: bench | blue
ATTR: orange | red

QUESTIONS:
1. [Object] Is there a bench in the image?
2. [Object] Is there an orange in the image?
3. [Attribute] Is the bench blue?
4. [Attribute] Is the orange red?
###

INPUT:
The wallet lies near a horse.

FACTS:
ENTITY: wallet
ENTITY: horse
SPATIAL: wallet | near | horse

QUESTIONS:
1. [Object] Is there a wallet in the image?
2. [Object] Is there a horse in the image?
3. [Spatial] Is the wallet near the horse?
###

INPUT:
Five drums.

FACTS:
COUNT: five | drums

QUESTIONS:
1. [Count] Are there exactly five drums in the image?
###

========================
NOW PROCESS THIS INPUT
========================

INPUT:
{PROMPT}

OUTPUT:
FACTS:
QUESTIONS:
###
"""
```

Figure 9: Introspector System Prompt Template used for Janus-Pro-7B models

Table 7: Qwen-Image training hyperparameters.

Name	Value
Learning rate	1×10^{-6}
Beta β	0
Group size G	16
Sampling steps T	10
Classifier-Free Guidance scale	4
Noise level α	1.2
Batch size	4
Gradient accumulation steps	8
Image resolution $h \times w$	512×512

Table 8: Bagel training hyperparameters.

Name	Value
Learning rate	1×10^{-4}
Beta β	0
Group size G	16
Sampling steps T	15
Classifier-Free Guidance scale	4.0
Noise level α	1.3
Batch size	6
Gradient accumulation steps	1
Image resolution $h \times w$	512×512

Table 9: Janus-Pro-7B training hyperparameters.

Name	Value
Learning rate	1×10^{-6}
Beta β	0.01
Group size G	8
Classifier-Free Guidance scale	5
Max gradient norm	1.0
Batch size	8
Gradient accumulation steps	2
Image resolution $h \times w$	384×384

Test-Time Training Detail and Cost In this section, we provide additional details on the test-time training procedure and its computational cost. To ensure fair comparisons across different reward and introspection configurations, we keep the number of test-time reinforcement learning steps identical for all configurations under the same backbone model and benchmark. All experiments are conducted on NVIDIA GPUs. Table 10 summarizes the number of test-time training steps used for each backbone model and benchmark.

Table 10: Number of test-time training steps used for each backbone model and benchmark.

Model	T2I-CompBench++	TIIF-Bench	DPG-Bench
Janus-Pro-7B	2,000	2,000	1,600
Bagel	500	500	200
Qwen-Image	500	500	200

We further report the full test-time training cost for each model and benchmark in Table 11. The total number of generated images and forward passes are computed as

$$\#\text{Images} = \text{steps} \times \text{batch size} \times \text{gradient accumulation} \times G, \tag{8}$$

$$\#\text{Forward passes} = \text{steps} \times \text{batch size} \times \text{gradient accumulation} \times (2G + 1), \tag{9}$$

where G denotes the number of candidate generations per prompt. For each prompt, one forward pass is used for rubric construction, while G forward passes are used for image generation and another G forward passes are used for scoring.

Table 11: Test-time training cost for each backbone model and benchmark.

Model	Benchmark	Test-Time Training Time (GPU-hours, 8 GPUs)	#Images	#Forward passes
Janus-Pro-7B	TIIF-Bench	34.4	256,000	544,000
	T2I-CompBench++	34.4	256,000	544,000
	DPG-Bench	27.5	204,800	435,200
Qwen-Image	TIIF-Bench	11	256,000	528,000
	T2I-CompBench++	11	256,000	528,000
	DPG-Bench	4.8	102,400	211,200
Bagel	TIIF-Bench	53	48,000	105,600
	T2I-CompBench++	53	48,000	105,600
	DPG-Bench	22.6	19,200	39,600

The reported cost corresponds to running **Meta-TTRL** until reward convergence. Since TTRL for T2I generation remains largely underexplored, we intentionally provide a full characterization of its optimization behavior and computational overhead. Importantly, the full convergence cost should not be interpreted as the required cost for every inference query. As shown in Table 12 and Figure 5, substantial performance gains already emerge at early training steps. Moreover, the adaptation cost of Meta-TTRL can be amortized as a one-time investment: once the model has been adapted, the improved model can be reused for future inputs without re-incurring the same optimization cost. This distinguishes **Meta-TTRL** from TTS methods such as Best-of-N, which must repeatedly perform costly sampling and ranking for every new prompt, even when prompts are similar.

To contextualize the computation cost, we additionally compare Meta-TTRL with a strong test-time scaling baseline, Best-of-N (BoN). We set $N = G = 8$, ensuring the same candidate budget per prompt, and use UnifiedReward [49] for ranking. Under the standard T2I-CompBench++ protocol with 10 samples per prompt, Meta-TTRL demonstrates a more favorable performance–efficiency trade-off than BoN. For example, Meta-TTRL with only 200 training steps already outperforms BoN in overall score, achieving 43.64 compared to 41.82, while using approximately $7\times$ fewer forward passes, 56.8K versus 384K. With increased test-time training compute, Meta-TTRL further improves to 48.90, suggesting that the method scales effectively with additional optimization.

Table 12: Efficiency comparison on T2I-CompBench++ using Janus-Pro-7B. Best-of-N uses $N = G$ candidates per prompt and UnifiedReward for ranking. "Time" denotes the average inference or test-time training cost measured in GPU-hours on 8 NVIDIA GPUs.

	3D			2D								
Method	Spatial	Color	Complex	NonSpatial	Numeracy	Shape	Spatial	Texture	Overall ↑	Time ↓	#Images	#Forward passes
Vanilla	27.37	52.02	37.98	31.27	44.50	32.72	15.73	40.39	35.25	1.2	24,000	24,000
+ BoN (N=8)	30.11	67.75	38.77	**31.44**	50.75	41.57	20.61	53.55	41.82	15.8	192,000	384,000
+ Meta-TTRL (200 steps)	<u>34.90</u>	<u>74.35</u>	<u>38.49</u>	<u>31.40</u>	<u>50.89</u>	<u>40.50</u>	<u>24.19</u>	<u>54.36</u>	<u>43.64</u>	4.6	49,600	56,800
+ Meta-TTRL (2000 steps)	**37.51**	**79.33**	**38.88**	31.35	**54.08**	**50.10**	**32.46**	**67.52**	**48.90**	35.6	280,000	568,000

C Pseudo code of Meta-TTRL framework

Algorithm 1 Meta-TTRL Framework

Input: Test prompt $x \in \mathcal{X}$
Initialize: Introspector $\theta \leftarrow \phi_0$ (frozen initialization), generator π_ϕ, rubric schema, maximum steps T
Initialize step counter $t = 0$
while $t < T$ and reward has not converged **do**
  Step 1: Meta-level Rubric Construction.
  Construct rubric $\mathcal{Q}(x)$ from the rubric schema for x.
  Step 2: Object-level Text-to-Image Generation.
  Sample candidates $\{y_1, y_2, \ldots, y_G\} \sim \pi_\phi(\cdot \mid x)$
  Step 3: Meta-level Intrinsic Monitoring.
  Evaluate candidates by introspector θ:
  for each candidate y_i **do**
    Compute score $s(y_i)$ for each rubric criterion
    Aggregate scores to obtain intrinsic reward $r(x, y_i)$
  end for
  Step 4: Meta-to-Object Policy Control via Intrinsic Signal.
  Control the generator via GRPO:
  Compute relative advantage A_i
  Form the GRPO objective $\mathcal{J}(\phi)$ in Eq. 6
  Update $\phi \leftarrow \phi + \eta \nabla_\phi \mathcal{J}(\phi)$
  Increment step counter: $t \leftarrow t + 1$
end while

D More Experiment Results

Table 13: Benchmarking on all categories with proposed metrics T2I-Compbench++ [21].

Model	Color	Shape	Texture	2D-Spatial	3D-Spatial	Numeracy	Non-Spatial	Complex
	B-VQA	B-VQA	B-VQA	UniDet	UniDet	UniDet	CLIP	3-in-1
Stable XL [39]	0.5879	0.4687	0.5299	0.2133	0.3566	0.4991	0.7673	0.7817
Pixart-α-ft [7]	0.6690	0.4927	0.6477	0.2064	0.3901	0.5032	0.7747	0.7823
DALLE·3 [5]	0.7785	0.6205	0.7036	0.2865	0.3744	0.5926	0.7853	0.7927
Stable v3 [12]	0.8132	0.5885	0.7334	0.3200	0.4084	0.6174	0.7782	0.7919
Qwen-Image [51]	0.8280	0.5986	0.7451	0.4430	0.4536	0.7582	0.3161	0.3838
BAGEL [51]	0.8081	0.5794	0.7154	0.3403	0.3778	0.6552	0.3111	0.3778
Janus-Pro-7B [8]	0.5202	0.3272	0.4039	0.1573	0.2737	0.4450	0.3127	0.3798
Qwen-Image + Meta-TTRL	0.8450	0.6242	0.7677	0.4659	0.4611	0.7677	0.3185	0.3848
BAGEL + Meta-TTRL	0.8269	0.6194	0.7455	0.3808	0.4369	0.6999	0.3128	0.3879
Janus-Pro-7B + Meta-TTRL	0.7933	0.5010	0.6752	0.3246	0.3751	0.5408	0.3135	0.3888

Table 14: **Quantitative evaluation results of instruct-following capability on TIIF testmini [50] (GPT-4o as the evaluation).** we report the metrics as presented in the official paper.

Model	Overall		Basic Following								Advanced Following												Designer	
			Avg		Attribute		Relation		Reasoning		Avg		Attribute +Relation		Attribute +Reasoning		Relation +Reasoning		Style		Text		Real World	
	short	long	short	long	short	long	short	long	short	long	short	long	short	long	short	long	short	long	short	long	short	long	short	long
FLUX.1-dev [27]	71.09	71.78	83.12	78.65	87.05	83.17	87.25	80.39	75.01	72.39	65.79	68.54	67.07	73.69	73.84	73.34	69.09	71.59	66.67	66.67	43.83	52.83	70.72	71.47
FLUX.1-Pro [27]	67.32	69.89	79.08	78.91	78.83	81.33	82.82	83.82	75.57	71.57	61.10	65.37	62.32	65.57	69.84	71.47	65.96	67.72	63.00	63.00	35.83	55.83	71.80	68.80
DALL-E 3 [5]	74.96	70.81	78.72	78.50	79.50	79.83	80.82	78.82	75.82	76.82	73.39	67.27	73.45	67.20	72.01	71.34	63.59	60.72	89.66	86.67	66.83	54.83	72.93	60.99
SD3.5-large [12]	71.15	66.96	78.34	79.56	79.50	76.50	80.96	83.21	72.46	78.71	67.67	61.18	66.46	61.89	73.53	74.15	60.03	61.53	73.33	63.33	70.52	42.52	64.43	66.39
PixArt-Σ [7]	62.00	58.12	70.66	75.25	69.33	78.83	75.07	77.32	67.57	69.57	57.65	49.50	65.20	56.57	66.96	61.72	66.59	54.59	83.33	70.00	1.83	1.83	62.11	52.41
Show-o [53]	59.72	58.86	73.08	75.83	74.83	79.83	78.82	78.32	65.57	69.32	53.67	50.38	60.95	56.82	68.59	68.96	66.46	56.22	63.33	66.67	3.83	2.83	55.02	50.92
T2I-R1 [23]	66.84	66.89	77.95	78.96	83.11	74.67	81.04	81.33	71.00	79.82	63.43	66.45	67.31	70.53	63.50	66.60	63.91	68.92	63.33	56.67	35.95	30.77	72.39	72.69
Qwen-Image [51]	83.45	82.48	83.96	83.82	84.50	89.50	87.53	82.16	79.86	79.81	75.44	78.11	76.99	78.13	72.14	79.40	72.03	74.46	96.67	86.67	95.48	88.24	85.82	83.96
BAGEL [10]	71.65	70.32	80.97	80.76	83.50	80.50	81.17	83.42	78.25	78.38	70.08	68.22	73.72	76.24	66.51	66.48	72.79	65.00	80.00	76.67	38.01	35.29	70.90	70.90
Janus-Pro-7B [8]	64.41	66.94	80.50	78.00	72.94	86.03	75.12	74.60	60.42	67.64	67.28	73.21	59.44	67.31	57.97	68.24	36.65	25.34	73.13	66.42	56.67	63.33	73.13	66.42
Qwen-Image + Meta-TTRL	85.28	84.67	85.32	85.29	87.50	92.50	87.13	86.09	81.33	77.29	78.44	79.31	76.90	79.40	74.43	79.60	79.35	75.85	96.67	90.00	97.29	93.21	86.94	88.06
BAGEL + Meta-TTRL	75.98	74.59	83.15	83.28	87.50	88.00	85.74	84.99	76.21	76.86	73.05	74.25	75.03	76.05	67.60	72.27	77.67	77.69	76.67	66.67	60.18	55.66	77.24	73.13
Janus-Pro-7B + Meta-TTRL	71.42	70.74	77.78	80.28	79.50	79.50	78.66	81.52	75.17	79.82	68.83	69.18	72.16	72.50	67.38	69.72	68.02	68.21	66.67	70.00	62.44	45.25	72.76	70.15
GPT-4o [22]	89.15	88.29	90.75	89.66	91.33	87.08	84.57	84.57	96.32	97.32	88.55	88.35	87.07	89.44	87.22	83.96	85.59	83.21	90.00	93.33	89.83	86.83	89.73	93.46

E More Qualitative Comparisons

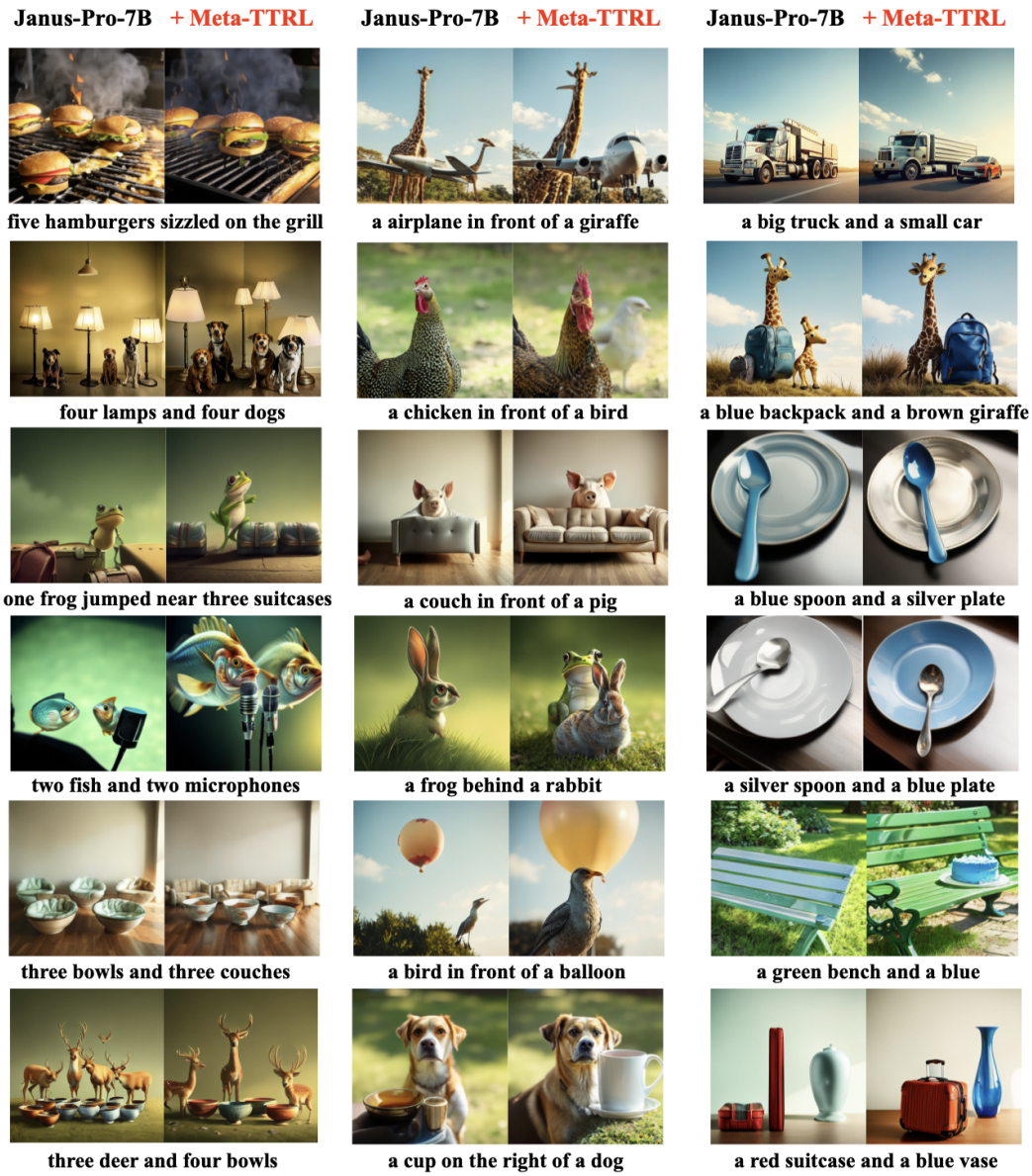


Figure 10: Qualitative comparison of T2I generation results on Janus-Pro-7B.

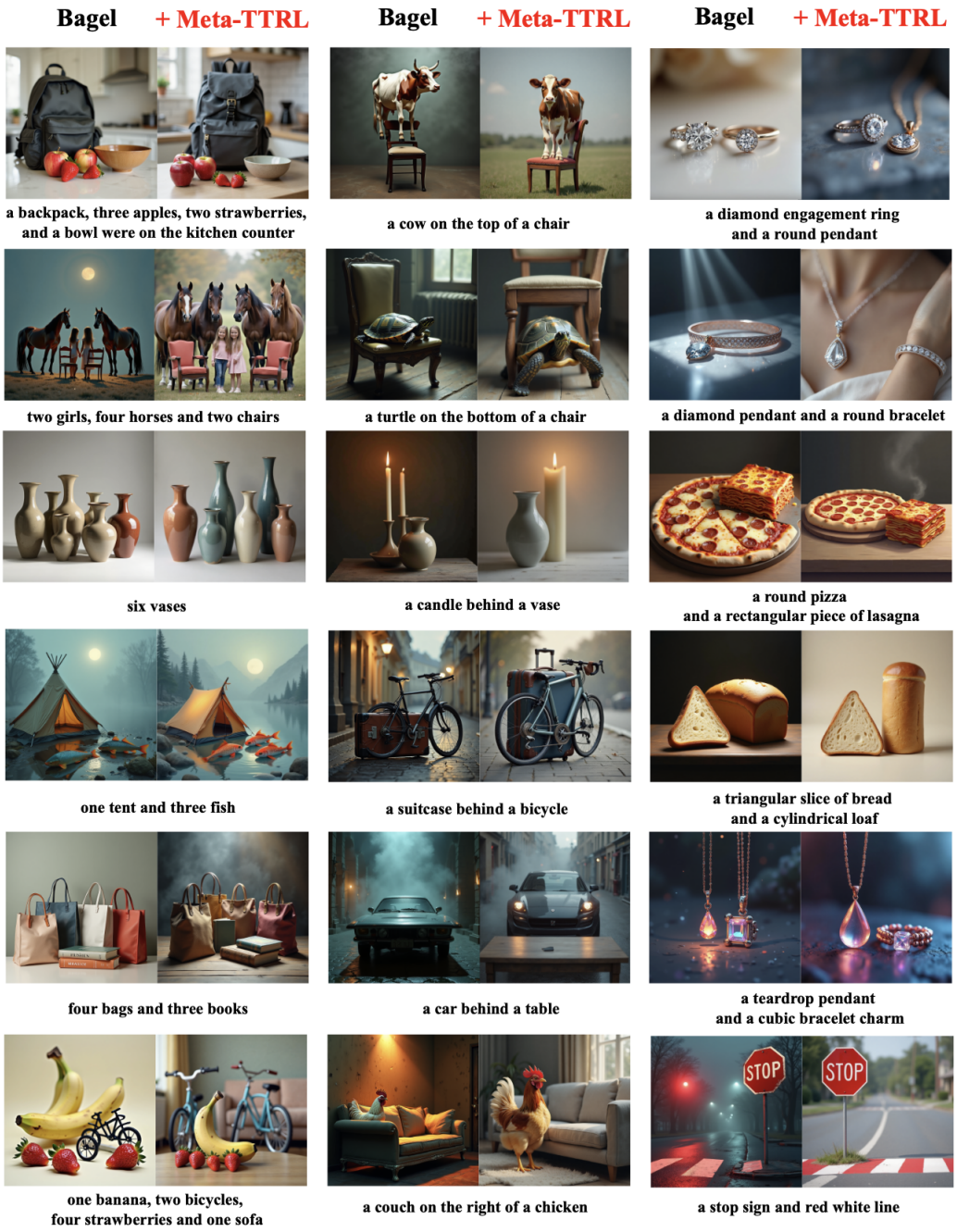


Figure 11: Qualitative comparison of T2I generation results on Bagel.



Figure 12: Qualitative comparison of T2I generation results on Qwen-Image.

F Training Dynamics of Meta-TTRL

To gain deeper insights into the optimization behavior of **Meta-TTRL**, we analyze the training dynamics during test-time optimization. We track the evolution of the GRPO reward and the corresponding T2I-CompBench++ average score. As depicted in Figure 13, both the reward and the benchmark performance show steady improvement throughout the optimization process. This closely synchronized growth suggests that **Meta-TTRL** optimizes meaningful feedback signals, avoiding the reward hacking often observed in reinforcement learning scenarios. The consistent increase in both reward and performance highlights **Meta-TTRL**'s capability to achieve sustained *self-improvement* during test-time optimization.

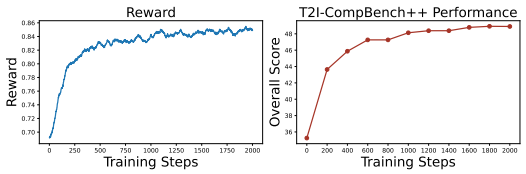


Figure 13: Evolution of GRPO reward and T2I-CompBench++ score during **Meta-TTRL** test-time optimization, showing synchronized improvement.

G Broader Impacts

This work introduces **Meta-TTRL**, a test-time reinforcement learning framework for text-to-image generation. Positive societal impacts include improved generative modeling capabilities and more accurate compositional image synthesis, which can benefit research and applications in digital content creation, design, and accessibility. Potential negative impacts include misuse for generating deepfake content, misinformation, or other malicious synthetic media. We note that these risks are inherent to generative AI technologies and not specific to our method.

H Safeguards

To mitigate potential misuse, the framework is designed for controlled access and requires model parameter availability, which inherently limits use on closed-source models. Upon code release, we plan to include usage guidelines, recommended ethical practices, and monitoring instructions to ensure responsible adoption.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract and introduction clearly state the Meta-TTRL framework, its self-improving capability, metacognitive synergy, and empirical gains across UMMs, matching the contributions in experiments.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: The paper explicitly notes limitations, such as the need for access to model parameters, restricting applicability to closed-source models

   Guidelines:

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

3. **Theory assumptions and proofs**

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

Answer: [Yes]

Justification: All key formulas (policy optimization, GRPO updates) are fully defined with assumptions, and derivations are provided to support theoretical claims.

Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: Experimental setup, benchmarks, models, metrics, and test-time training steps are fully described, allowing reproduction of main results

   Guidelines:

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

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

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

Answer: [No]

Justification: Code release upon acceptance. The paper does not currently provide links to code, although all methods and hyperparameters are described in detail.

Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: Training and test details, hyperparameters, optimizer settings, and GRPO parameters are provided in tables and appendices.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: Results are reported with multiple model comparisons, subdimension metrics, and figures/tables showing performance gains.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: GPU type, total compute hours, batch size, and forward pass counts are reported for each experiment.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: Experiments conform to NeurIPS ethics guidelines; no unethical data use or procedures are involved.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: The paper discusses positive impacts (improved T2I generation) and potential negative impacts (misuse for deepfakes), and suggests mitigation strategies.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.

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

11. **Safeguards**

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

    Answer: [Yes]

    Justification: Method limitations imply safeguards; closed-source models are restricted, and responsible usage is highlighted.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: All datasets and models used are cited with original papers, versions, and licenses indicated.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [Yes]

    Justification: The paper introduces new assets in the form of Meta-TTRL model outputs, the introspector rubric templates used for intrinsic evaluation, and the procedures for test-time policy optimization. These assets are clearly documented in the text, figures, and appendices, providing sufficient information for other researchers to understand and potentially replicate the evaluation and generation workflow.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: No human subjects or crowdsourcing experiments are conducted.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: Not applicable as no human participants were involved.

    Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: The paper explicitly reports that GPT-4o and Qwen3 are used as external introspectors in the experiments to provide monitoring signals for test-time reinforcement learning. These models are used only for evaluation and comparison purposes, separate from the core Meta-TTRL framework, which relies on intrinsic introspection within the base UMMs. This distinction ensures that the main methodological contributions do not depend on proprietary LLMs, and their role is clearly described for transparency and reproducibility.

    Guidelines:

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