CompJudge: Fine-Grained Comparative Evaluation using Multimodal LLM for Subject-Driven Generation

Nam Hyeon-Woo*
Department of Electrical Engineering
POSTECH
hyeonw.nam@gmail.com

Wenbin Ouyang
Amazon
wenbinoy@amazon.com

Ciprian Corneanu
Amazon
cipriancorneanu@gmail.com

Abstract

Subject-driven generation (SDG) evaluation using multi-modal LLMs (MLLMs) provides better interpretability and stronger alignment with human judgment than traditional embedding-based methods. However, MLLM-as-a-Judge scores are limited in differentiating between error and error-free SDG images. In this paper, we propose CompJudge, which overcomes these limitations by introducing comparative capabilities to MLLM-as-a-Judge and recalibrating scores based on comparison results. To rigorously evaluate the correctness of SDG evaluators, we introduce CompILIAS, a diagnostic benchmark of triplets (reference, identity-preserving image, identity-degraded image) with verified binary ground-truth labels indicating whether the SDG image preserves the reference identity. We demonstrate improvements of SDG evaluation over existing approaches: 8.9pp on CompILIAS and 10.62–15.98pp on the DreamBench++ dataset. These improvements consistently generalize across diverse MLLM backbones and across subject categories.

1 Introduction

Subject-driven generation (SDG) aims to synthesize novel images while preserving the identity of a specific subject from a reference image. SDG has diverse applications, including virtual try-on [24], story visualization [46], personalized video generation [9], and facial editing [17]. High-fidelity subject consistency is critical. For example, in domains like advertising, misrepresentation of a product can damage brand credibility, erode consumer trust, and incur legal liabilities. Thus, accurate evaluation of subject identity preservation is essential.

Traditional SDG evaluation relies on embedding-based metrics such as CLIP [45, 26] and DINO [8]. While widely used, these metrics [51, 27] often fail to capture the crucial fine details of subject identity needed to assess true target fidelity, such as text or logos. To overcome these limitations, using MLLMs as evaluators (MLLM-as-a-Judge) has emerged as a promising alternative, leveraging their visual understanding capabilities to score SDG images [43].

MLLM-as-a-Judge uses specific criteria designed for evaluation [43, 25, 28]. MLLM-as-a-Judge scores images based on whether they satisfy predefined attributes (*i.e.*, shape, color, texture). While structured, this absolute scoring method suffers from a measurement limitation: the ceiling effect [16]. Because state-of-the-art SDG models meet the criteria, their scores cluster near the upper end of the score range, making it difficult to discriminate between modern high-performing models. Furthermore,

*Work done during an internship at Amazon.

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

the absolute scores obtained from MLLM-as-a-Judge are unreliable for ranking recent SDG models with subtle yet meaningful differences in identity preservation and visual fidelity.

To address these challenges, we propose a novel SDG evaluation method that replaces absolute scoring with pairwise comparative scoring within MLLM-as-a-Judge. Our method operates effectively as a modular plugin for existing MLLM-as-a-Judge. We introduce two key components: *Comparative Analysis Module* and *Judge Refinement Module*. Instead of scoring an image in isolation, our method compares SDG images to determine relative fidelity as shown in Fig. 1.

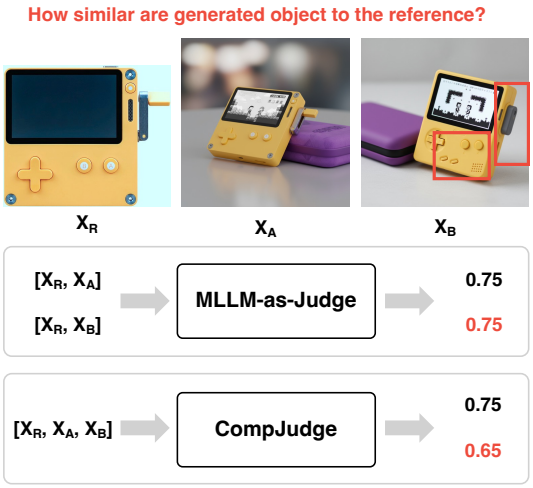


Figure 1: MLLM-as-a-Judge versus our CompJudge. X_R is the reference, X_A is an error-free subject, and X_B contains fidelity errors. CompJudge refines the scores by explicit pairwise comparison of $X_{A,B}$ against X_R.

Specifically, the Comparative Analysis Module identifies deviations between the reference and generated images and determines which SDG image in a pair exhibits greater fidelity. Subsequently, the Judge Refinement Module translates these qualitative judgments into calibrated numerical scores using the Bradley-Terry model [7], enabling precise quantitative ranking. To maximize the MLLM's discriminatory power, we employ TextGrad [63], an automatic prompt-optimization framework, to identify an effective prompt. We formulate *Criticism Loss*, a textual loss function tailored for this comparative task of fine-grained identity discrimination.

Furthermore, to rigorously measure the correctness of SDG evaluators on subtle identity shifts, we introduce CompILIAS, a diagnostic benchmark with verified binary ground truth. We evaluate CompJudge on CompILIAS and the DreamBench++ dataset. The results demonstrate that our comparative approach achieves a state-of-the-art accuracy of 85.1%. These gains generalize across diverse MLLM backbones and across all subject categories. CompJudge not only aligns better with human comparative judgment but also effectively mitigates the ceiling effect by distinguishing fine-grained differences among high-performing SDG models. In summary, we present three jointly-designed contributions for fine-grained SDG evaluation that together form a non-trivial advance for a community where measurement reliability now bottlenecks the comparison of modern generators:

- CompJudge, an SDG evaluator method composed of a Comparative Analysis Module and a Judge Refinement Module that turn MLLM pairwise judgments into calibrated, discriminative scores;
- Prompt-optimization procedure tailored for comparative SDG evaluation, including a Criticism Loss that sharpens the evaluator's fine-grained discriminative power;
- CompILIAS, a diagnostic benchmark of triplets with verified binary ground truth, providing a rigorous testbed for measuring evaluator correctness on fine-grained identity discrimination.

2 Related work

Subject Driven Generation (SDG). Recent generative models have demonstrated remarkable capabilities in high-quality visual generation [48, 44, 42, 31]. Diffusion models, in particular, have proven successful in this domain, catalyzing active research in image editing and enabling unprecedented control over visual content synthesis [65, 22, 2, 23]. The primary goal of SDG is to generate edited images while preserving the identity of a given subject [13, 35, 14, 62, 34, 12, 56, 10, 40, 10, 39, 64, 57]. Gal et al. [19] proposed Textual Inversion, which discovers the textual token corresponding to a specific subject identity, while Ruiz et al. [50] introduced DreamBooth, which fine-tunes diffusion models to inject subject-specific identity knowledge. Alternative methods [20, 47] directly extract subject identity information from visual encoders and inject it into the diffusion model without requiring additional fine-tuning, exemplified by Qwen Image Edit [58] and Kontext [6]. More recently, NanoBanana [15] has emerged as another effective SDG work, gaining widespread adoption in the community due to its strong performance. In this work, we address the challenge of SDG evaluation.

SDG Evaluation. Evaluators for image generation are generally categorized into two types: embedding-based evaluation [26, 41, 53] and MLLM-as-a-Judge frameworks [28, 43, 25]. Embedding-based evaluation extracts feature vectors from encoders and computes similarity metrics between the subject and SDG images. While effective, this approach faces fundamental limitations in reliably measuring fine-grained object identity between images. Fine details are lost while constructing embeddings. For example, visual embeddings often fail to capture text rendering errors.

To address these limitations, MLLM-as-a-Judge frameworks have emerged alongside recent advances in MLLMs. For MLLM-as-a-Judge, the evaluation criteria are provided to the model as a textual description, and it then assesses the images. MLLMs have demonstrated robust visual understanding capabilities, including OCR, visual reasoning, and detailed analysis [3, 55, 5, 4, 37, 1, 21, 15]. In the context of SDG evaluation, Peng et al. [43] have proposed DreamBench++, defining critical evaluation axes (*e.g.*, color, texture, text) where MLLMs generate scores based on an analysis of each dimension. This provides a more comprehensive assessment framework for SDG. However, we observe that MLLM-as-a-Judge cannot correctly rank SDG models in fine-grained comparisons. In this work, we advance prior MLLM-as-a-Judge approaches by introducing comparative analysis modules and the optimization method to produce more reliable scores for comparison scenarios.

Comparative Evaluators. Comparative analysis is widely utilized across various domains, including data annotation [11] and model analysis [18, 52]. In labeling tasks, worker bias often leads to inconsistent scoring among annotators. To mitigate this, Chen et al. [11] have leveraged pair-wise comparison to obtain continuous values. A similar challenge arises in evaluating generative models: since generation quality is difficult to quantify with absolute values, pair-wise comparison (AB Testing) is often adopted to derive metrics such as win rates. Jiang et al. [30] have proposed an evaluation platform for generative models and introduced prompts to mimic human voting. However, they have observed that MLLMs achieve lower performance due to limited instruction-following and reasoning ability. Unlike previous works, we focus specifically on SDG evaluation, where preserving subject identity is important. We introduce CompILIAS, a diagnostic benchmark of triplets for evaluating SDG evaluators, and maximize the evaluator's discriminative power through automatic prompt optimization with our *Criticism Loss*. By integrating comparative analysis into MLLM-as-a-Judge, we observe that MLLM-as-a-Judge can assign the correct judge (Fig. 1).

3 Method

3.1 Motivation

The *ceiling effect* describes a measurement limitation where the majority achieve near-maximum scores, thereby diminishing the discriminative power [16]. While previous MLLM-as-a-Judge methods have effectively evaluated SDG models across varying quality levels, they encounter this ceiling effect when assessing recent high-performance models. To validate this phenomenon, we use DreamBench++ to evaluate SDG models, IP Adapter Plus [62], Kontext [6], and Nano Banana [15].

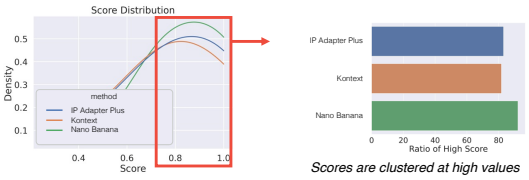


Figure 2: Score plots of DreamBench++. The chart shows that the scores of both outdated and recent models are clustered at high values (≥ 0.75). Due to the saturated scores of naive metrics, modern SDG methods are less distinguishable.

We present the score distributions in Fig. 2. The scores for both outdated and recent models are heavily clustered at the upper bound of the scale. A direct comparison between Kontext (2025) and IP Adapter Plus (2023) reveals a counter-intuitive result: the older model achieves comparable or even higher scores, contradicting visual inspection. This raises a fundamental question: *Can we validly conclude that IP Adapter Plus + SDXL outperforms Kontext?* Our human study suggests otherwise (Sec. 4.5).

To understand this measurement failure, let y denote the actual score of an SDG image X given a reference X_R. Without loss of generality, we define the standard MLLM-as-a-Judge process as a function f that maps X to a discrete score set S [43]:

$$f(X|X_R) \in S, \quad \text{where } S = \{0, 1, 2, 3, 4\}. \tag{1}$$

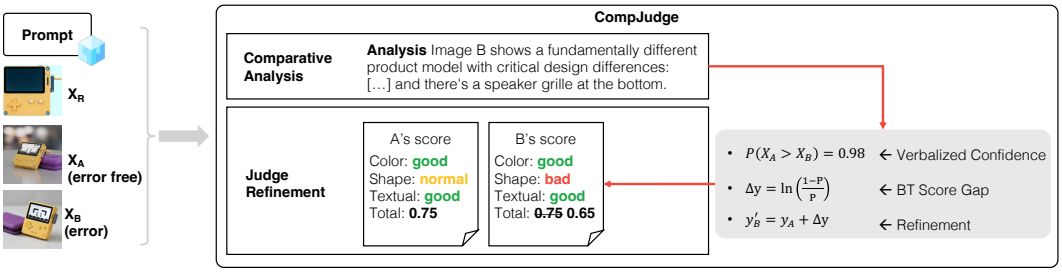


Figure 3: CompJudge is composed of Comparative Analysis and Judge Refinement Modules that enable an MLLM to perform comparative analysis. Given the reference X_R and two images $X_{A/B}$, DreamBench++ assigns identical scores even though X_B should rank below X_A. CompJudge resolves this via pairwise comparison and Bradley-Terry score refinement.

Although MLLMs can be prompted to output continuous values, their internal reasoning often collapses into discrete categories (or Likert scales) rather than a true continuum. This limitation concerns response-based scoring, in which the judge reports its score as text. Logit-based scoring such as Q-Bench [59] requires access to token probabilities, and we compare it in Sec. E.4. This introduces a critical limitation, namely indistinguishability despite qualitative differences. Mathematically, the equality of evaluator scores does not imply the equality of actual scores:

$$f(X_A|X_R) = f(X_B|X_R) \implies y_A = y_B. \tag{2}$$

We measured the rate of *false equivalence* on our dataset (Sec. 3.3), where each triplet pairs an identity-preserving image with an identity-degraded one so that $y_A \neq y_B$ holds by construction. False equivalence refers to cases where the evaluator collapses this latent difference into identical scores ($f(X_A) = f(X_B)$). Using DreamBench++, a widely adopted MLLM-as-a-Judge for SDG, false equivalence accounts for 91.18% of its incorrect predictions, indicating that the dominant failure mode of naive MLLM scoring is the lack of granularity rather than outright rank inversion.

When these tied scores accumulate across samples, the aggregate model ranking can be inverted. Let the aggregate true scores of Model A and B be $E[y_A]$ and $E[y_B]$, and their evaluator scores be $E[f(X_A|X_R)]$ and $E[f(X_B|X_R)]$. The evaluator's order does not guarantee the true order:

$$E[f(X_A|X_R)] > E[f(X_B|X_R)] \implies E[y_A] > E[y_B]. \tag{3}$$

Consider an evaluator that rounds scores to the nearest integer. For $\mathbf{y}_A = [1.5, 1.5, 1.5]$ and $\mathbf{y}_B = [2.0, 2.0, 1.4]$, the true scores are $E[y_A] = 1.5$ vs. $E[y_B] = 1.8$ (Model B is better), but the evaluator produces $[2, 2, 2]$ vs. $[2, 2, 1]$, yielding $E[f(X_A|X_R)] = 2$ vs. $E[f(X_B|X_R)] = 1.67$, incorrectly ranking Model A higher. Note that this inversion arises from accumulated ties at $f(\cdot)$; resolving the within-tie identification therefore directly mitigates such ranking errors.

How can we mitigate these inherent limitations? We propose shifting from absolute scoring to pairwise comparison. We introduce a comparator c defined as:

$$c(X_A, X_B|X_R) = \mathbf{1}(y_A > y_B) - \mathbf{1}(y_B > y_A). \tag{4}$$

Based on this, we define a refined evaluator g that leverages relative differences:

$$g(X_A|X_R) - g(X_B|X_R) \propto \lambda \cdot c(X_A, X_B|X_R), \tag{5}$$

where λ is a scaling hyperparameter, fixed to $\lambda = 1$ throughout this work. By directly estimating the probability of one image being better than another, we address the measurement limitation of absolute scoring. Our goal is to enhance the MLLM-as-a-Judge with this comparative capability, enabling precise and order-preserving evaluations.

3.2 CompJudge

CompJudge is composed of two components: *Comparative Analysis Module* ($c(\cdot,\cdot)$) and the *Judge Refinement Module* ($g(\cdot)$). The Comparative Analysis Module performs a pairwise comparison. The Judge Refinement Module utilizes the outcome of this comparison to recalibrate the scores.

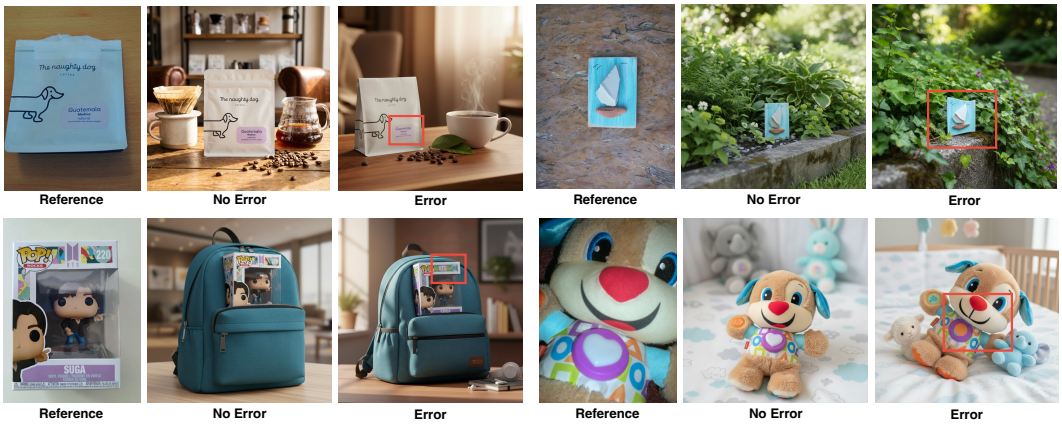


Figure 4: Representative triplets from CompILIAS. Each example shows a reference X_R alongside two SDG images $\{X_A, X_B\}$, one identity-preserving and one identity-degraded. The two outputs differ only by fine-grained artifacts, such as minor text rendering errors, subtle logo distortions, or small texture shifts, requiring evaluators to perform fine-grained discrimination rather than detect obvious failures.

Comparative Analysis Module. Let X_R denote the reference subject image, and let X_A and X_B represent SDG images. Suppose that the initial evaluator assigns identical scores to both images ($f(X_A|X_R) = f(X_B|X_R)$). Leveraging the multi-image processing capabilities of MLLMs, we feed the triplet (X_R, X_A, X_B) into the model. The MLLM scrutinizes fine-grained discrepancies between the reference X_R and the generated candidates (X_A, X_B) to determine which image exhibits superior preservation of subject identity. As shown in Fig. 3, the module produces an Analysis that describes the observed identity-preservation errors.

Judge Refinement Module. After the Comparative Analysis Module identifies the superior candidate (*i.e.*, X_A outperforms X_B), we quantify the certainty of this judgment using verbalized confidence [36, 61, 54]. We prompt the MLLM to output a probability score $P(X_A \succ X_B)$, representing the likelihood that X_A is superior to X_B given X_R. P serves as a probabilistic estimate of c, providing a continuous signal for the BT-based refinement below. This use of P is consistent with the limitation discussed in Sec. 3.1. P is not an absolute score of a single image but the confidence of a relative decision made with the competing candidate in view, and it enters the refined score only through the BT gap below. We validate this confidence against empirical correctness in Sec. 4.3. We then transform this probability into a relative score adjustment using the BT model, a standard probabilistic framework for pairwise comparisons. Letting $\Delta y = y_B - y_A$ denote the score gap, the BT model relates the preference probability to the gap as:

$$P\left(X_A \succ X_B\right) = \frac{1}{1 + \exp\left(\Delta y\right)}. \tag{6}$$

From this equation, the difference in latent scores can be derived as:

$$\Delta y = \ln\left(\frac{1 - P\left(X_A \succ X_B\right)}{P\left(X_A \succ X_B\right)}\right), \tag{7}$$

where $P\left(X_A \succ X_B\right)$ is clipped to $[\varepsilon, 1-\varepsilon]$ with a small $\varepsilon > 0$ for numerical stability. Consequently, we compute the refined score y'_B by adjusting the baseline score y_A with the difference:

$$y'_B = y_A + \Delta y. \tag{8}$$

Refinement is applied only among images that share the same initial score, and the accumulated adjustment within such a group is bounded so that refined scores never cross into an adjacent score level.[2] Through this process, we convert qualitative preferences into a calibrated numerical difference. As shown in Fig. 3, the score is refined; the full algorithm is presented in Algorithm 1. Comparisons follow a sort-based procedure, requiring $\mathcal{O}(N \log N)$ pairwise queries in the number N of compared SDG models. Latency analysis is provided in Sec. E.2.

[2]We use $\varepsilon = 10^{-4}$. Initial scores are placed on a scale where adjacent levels of S differ by 10, and if the accumulated adjustment in a group reaches 10, all adjustments in the group are rescaled to sum to $\ln((1-\varepsilon)/\varepsilon) \approx 9.21$.

Prompt Optimization. To maximize the discriminative power of our comparative analysis, we optimize the prompt using TextGrad [63], an automatic prompt optimization framework. The optimization process consists of three steps: (1) *Prediction:* Given a triplet (X_R, X_A, X_B) and the current prompt p_t, the MLLM generates a comparative response. (2) *Loss and Feedback:* An external MLLM evaluates the prediction against the ground-truth label to compute a textual "loss" and generates feedback for improvement. (3) *Update:* The optimizer MLLM updates the prompt from p_t to p_{t+1} based on the feedback. As illustrated in Fig. 5, the loss function is specifically tailored to the comparative analysis task and is computed on a held-out subset of CompILIAS (Sec. 3.3) comprising 36 training and 20 validation triplets, disjoint from all evaluation sets used in our experiments. Note that this optimization is an offline process; the derived prompt remains static during the inference phase of SDG evaluation. The pseudo-code is provided in Algorithm 2, and further details are available in Sec. A of the appendix.

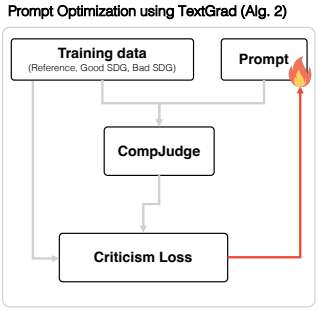


Figure 5: Prompt optimization with Criticism Loss.

3.3 CompILIAS, a diagnostic benchmark for SDG evaluators

We introduce CompILIAS, a *diagnostic benchmark* for measuring the correctness of SDG evaluators. Unlike standard SDG datasets, CompILIAS provides a controlled test set with verified binary ground truth. Each triplet $\{X_R, X_A, X_B\}$ contains one identity-preserving and one identity-degraded image, paired with a label $b = 1(X_A \succ X_B)$ where $1(\cdot)$ is the indicator function. This design enables direct measurement of whether an evaluator can distinguish error-free from error-containing images. CompILIAS therefore tests whether an evaluator detects verified identity errors. Its degradations are designed to mimic identity-error types observed in real SDG outputs, but we do not claim that they reproduce the precise appearance or frequency of natural failures. We evaluate real SDG outputs separately in Sec. 4.1.3.

Triplet Examples and Difficulty. Figure 4 shows representative triplets. The identity-degraded image differs from the identity-preserving image only by fine-grained artifacts, such as minor text rendering errors. These triplets capture how modern SDG models actually fail, namely with subtle deviations. Absolute-scoring metrics miss these deviations.

Annotation Quality and Hard Filtering. CompILIAS is the product of a deliberately strict filtering protocol. Each candidate triplet was independently labeled by two annotators, with a Cohen's Kappa of 0.6375 between annotators. To guarantee unambiguous ground truth, we then applied a strict consensus filter: any triplet flagged by even one annotator was excluded, resulting in a 72% elimination rate. The final benchmark contains 1,114 high-quality triplets; the reference images are sourced from the ILIAS instance-retrieval dataset [33].

Grounded in Real SDG Errors. The error categories covered by CompILIAS are not chosen arbitrarily; they are identified empirically. We performed a two-stage characterization of real SDG failure modes: (i) an in-house analysis by 30 internal evaluators (including scientists, advisors, product managers, engineers, and UX designers) on 1,000 SDG outputs, yielding eight fine-grained error categories; and (ii) a large-scale MTurk validation on two architecturally distinct generators (Nano Banana Pro and Flux 2), covering 1,498 products with four prompts each, with three workers per sample and approximately 18,000 total annotations ($1{,}498 \times 4 \times 3$), confirming that all eight categories appear on both SDG models with broadly similar distributions. CompILIAS provides a controlled dataset grounded in the actual failure modes observed in modern SDG outputs. This taxonomy determines which error types CompILIAS covers, not their relative frequency. Full per-category statistics and the MTurk study methodology are provided in Sec. B and Sec. B.3.

4 Experiments

This section reports our main empirical findings, covering pairwise comparison effectiveness against baselines (Sec. 4.1), comparison with ensemble aggregation (Sec. 4.2), calibration of verbalized probabilities (Sec. 4.3), prompt optimization analysis (Sec. 4.4), and ceiling effect mitigation (Sec. 4.5).

Table 1: Pairwise comparison accuracy on CompILIAS. Ours outperforms both embedding-based evaluators and MLLM-as-a-Judge. Lenient assigns 0.5 to ties, and Strict counts ties as incorrect. A single value is shown for evaluators that never produce ties.

Evaluator		Accuracy [%]	
Approach	Method	Lenient	Strict
Embedding	CLIP	60.95	
	DINO	62.83	
	DINOv2	61.67	
	DINOv3	66.15	
	LoftUp (DINOv2)	54.76	
MLLM	DreamBench++	76.17	56.19
	Verbal Score	76.70	57.18
	CompJudge (Ours)	**85.10**	

Table 2: Impact of other MLLM backbones. Regardless of MLLM backbones, ours improves MLLM-as-a-Judge.

Evaluation Method	MLLM	Accuracy [%]
DreamBench++	Claude 4.0	72.76
Verbal Score		69.97
CompJudge		**80.61**
DreamBench++	Claude 4.5	76.17
Verbal Score		76.70
CompJudge		**85.10**
DreamBench++	GPT 5 Mini	77.34
Verbal Score		73.65
CompJudge		**90.80**

The appendix further establishes prompt transferability across optimizer-backbone combinations (Sec. E.3), modular extension to prompt-following on CompILIAS-PF (Sec. E.7).

4.1 General effectiveness

For embedding-based baselines we use CLIP [45], DINO [8], DINOv2 [41], DINOv3 [53], and LoftUp [29]. For MLLM-as-a-Judge we adopt DreamBench++ [43], adapted to output a verbal score (*e.g.*, "good") instead of a numerical one. Unless noted, the MLLM is Claude Sonnet 4.5 with temperature 0, and the prompt is optimized once on Claude Sonnet 3.7 (Sec. A).

Evaluation protocol. We report pairwise comparison accuracy. For each triplet, the evaluator receives 1 point for identifying the identity-preserving candidate, 0 for the wrong choice, and 0.5 when it assigns equal scores to the two candidates. We assign 0.5 to ties because absolute-scoring evaluators like DreamBench++ are not designed for discriminative ranking on our challenging triplets. Unless otherwise stated, we use this lenient rule, which favors tie-prone absolute-scoring evaluators such as DreamBench++ because 0.5 equals the expected accuracy of random tie-breaking. Table 1 also reports a strict rule that counts every tie as incorrect. Because CompJudge refines the absolute scores of the base judge (DreamBench++) rather than replacing them, every image retains an absolute score, and model-level scores are their averages per SDG model, normalized to $[0, 1]$.

4.1.1 Pairwise comparison effectiveness

Table 1 reports pairwise comparison accuracy on CompILIAS. Even the strongest embedding-based evaluator, DINOv3, reaches only 66.15% and falls short of DreamBench++ (76.17%) and its Verbal Score variant (76.70%). *CompJudge achieves 85.10%, an 8.93pp gain over DreamBench++*. Under the strict rule, DreamBench++ drops to 56.19% while CompJudge remains at 85.10%, widening the gap to 28.91pp. CompJudge thus captures the fine-grained identity errors that absolute-scoring evaluators conflate. Is the gap due to DreamBench++ scoring each candidate separately? We give DreamBench++ the same triplet and require it to score the better candidate strictly higher. The gap remains at 9.42pp (Table 3).

Table 3: DreamBench++ given the same triplet with forced choice (Claude Sonnet 4.5).

Evaluator	A/B	B/A	Avg.
DreamBench++	80.25	71.10	75.68
CompJudge (Ours)	83.75	86.45	**85.10**

4.1.2 Backbone independence

In Table 2, we show the robustness of our approach to various underlying MLLM backbones. *Irrespective of the specific MLLM version employed, our method consistently delivers performance improvements*, demonstrating its model-agnostic effectiveness.

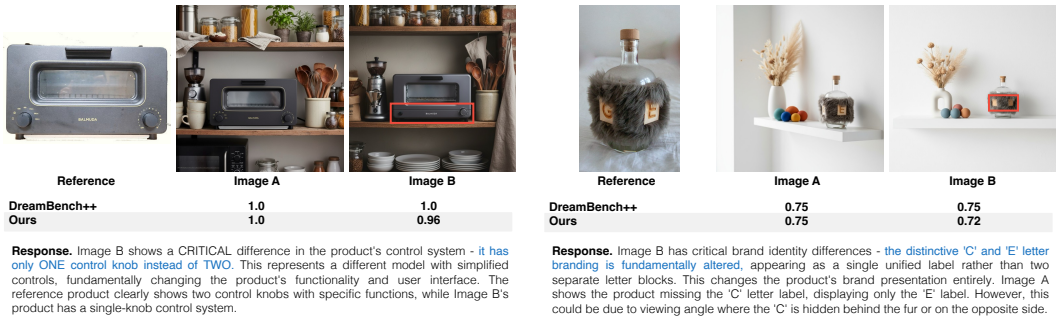


Figure 6: Qualitative comparison of scoring behaviors. While DreamBench++ assigns identical scores despite obvious artifacts, our method detects these defects through comparative analysis and refines the score, providing the reasoning for the adjustment.

Table 4: Pairwise comparison accuracy on DreamBench++. CompJudge consistently improves over DreamBench++ across all subjects.

Subject	DreamBench++	CompJudge
Animal	71.23	**87.21**
Human	81.97	**92.59**
Object	79.13	**91.43**

Table 5: Evaluation scores on the DreamBench++ dataset. Rank is shown in brackets.

SDG Method	Human	DreamBench++	CompJudge
IP Adapter Plus	0.6863 (1)	0.7624 (1)	0.7215 (1)
DreamBooth LoRA	0.5077 (2)	0.5440 (2)	0.4900 (2)
IP Adapter	0.4940 (3)	0.5094 (3)	0.4791 (3)
BLIP Diffusion	0.4556 (5)	0.4714 (5)	0.4756 (4)
DreamBooth	0.4623 (4)	0.4906 (4)	0.4324 (5)
Emu2	0.3623 (6)	0.4470 (6)	0.3965 (6)
Textual Inversion	0.2654 (7)	0.3030 (7)	0.2721 (7)

4.1.3 Alignment with human judgement

Because our prompt is optimized on CompILIAS, its held-out accuracy alone could overstate performance on natural SDG failures. The MLLM weights remain frozen, and the final prompt describes a general identity-comparison procedure without referring to the degradation operations used to construct CompILIAS (Sec. D). We therefore apply this prompt without modification to the DreamBench++ dataset, which is independently constructed from real SDG outputs with human labels. This includes the Animal and Human categories, although the prompt uses product-oriented wording. We evaluate pairwise comparison accuracy on a filtered subset where two human annotators agreed which generated image is closer to the reference. As Table 4 shows, CompJudge achieves substantial improvements on every subject category: +15.98pp on Animal, +10.62pp on Human, and +12.30pp on Object. These gains confirm that *CompJudge enhances alignment with human judgment in fine-grained comparisons*, and that the improvement transfers beyond the controlled CompILIAS construction.

A specific case illustrates how pairwise scoring captures human preferences. CompJudge matches the human and DreamBench++ ordering on most SDG models in Table 5; the one ranking change is that BLIP Diffusion moves above DreamBooth. Recomputing the pairwise win rate from raw DreamBench++ human scores reveals that *BLIP Diffusion wins 57.71% of head-to-head comparisons against DreamBooth*. The reversal therefore captures a comparative preference that absolute scoring smooths out, namely what humans actually choose in head-to-head comparisons.

4.1.4 Qualitative examples

In Fig. 6, we show comparison examples between CompJudge and DreamBench++. We observe that DreamBench++ assigns identical scores to image pairs, even when one clearly exhibits visible defects. This reflects that absolute scoring without comparison is prone to inconsistency, mirroring the cognitive biases often found in human annotation [11]. In contrast, *CompJudge effectively calibrates these initial scores to reflect the true quality difference*. Furthermore, the response explicitly includes the rationale for this refinement. The MLLM successfully identifies specific generation artifacts, providing interpretable feedback alongside the adjusted scores.

Table 6: Comparison with PoE-based ensemble aggregation on CompILIAS.

Method	Accuracy [%]
PoE (attributes)	73.88
PoE (backbone)	78.00
CompJudge (Ours)	**85.10**

Table 7: Prompting vs. image order. Comparison with GenAI [30]. A/B and B/A denote the order combination.

Prompt	MLLM	Accuracy [%]		
		A/B	B/A	Average
CompJudge Prompt	Claude 4.0	80.07	81.15	**80.61**
GenAI Prompt		60.23	85.19	72.71
CompJudge Prompt	Claude 4.5	83.75	86.45	**85.10**
GenAI Prompt		71.10	83.30	77.20

Table 8: Component ablation on CompILIAS (Claude Sonnet 4.5).

Configuration	Acc. [%]
Absolute scoring (DreamBench++)	76.17
+ Pairwise comparison (GenAI prompt)	77.20
+ TextGrad (binary loss)	78.50
+ Criticism Loss (CompJudge)	**85.10**

Table 9: Percentage of samples with high (≥ 0.75) and maximum ($= 1.0$) score.

SDG Method	Score ≥ 0.75		Score $= 1.0$	
	DreamBench++	Ours	DreamBench++	Ours
IP Adapter Plus	83.12	52.25	42.86	6.49
Kontext	81.82	61.04	35.06	23.38
Nano Banana	92.21	71.42	46.75	29.87

4.2 Comparison with ensemble baselines

A natural alternative to our BT-based refinement is to aggregate multiple MLLM judgments via Product-of-Experts (PoE), treating different evaluation attributes or different MLLM backbones as independent experts. We compare CompJudge against two PoE variants: (i) PoE over per-attribute scores (DreamBench++'s shape, color, and texture scores from Claude Sonnet 4.5), and (ii) PoE over per-backbone judgments (the three backbones in Table 2). For each pair (X_A, X_B), each expert outputs a score in $[0, 1]$, and we aggregate these scores across experts by their product. As shown in Table 6, both PoE variants underperform CompJudge, demonstrating that the BT-based refinement of pairwise comparisons captures discriminative signals that simple ensembling cannot.

4.3 Calibration of verbalized probabilities

Our Judge Refinement Module uses the verbalized probabilities produced by MLLM judges as inputs to the Bradley-Terry model. We measure the Expected Calibration Error (ECE) on CompILIAS and observe ECE $= 0.061$ for Claude 4.0 and ECE $= 0.108$ for Claude 4.5, well within the operating range required for BT-based ranking. Post-hoc calibration techniques such as ensembling or Platt scaling could further sharpen the probability estimates and remain a promising future direction.

4.4 Prompt optimization

To isolate the contribution of prompt design, we replace our optimized prompt with GenAI [30], a prompt designed for A/B testing, while keeping the method fixed. Table 7 reports the results.

Our optimized prompt outperforms GenAI by 7.9pp with Claude 4.5 (85.1% vs. 77.2%), and remains robust to the order of target images. As shown in Table 7, the gap between A/B and B/A orderings is only 1.08-2.70pp for our prompt versus 12.20-24.96pp for GenAI, roughly an order of magnitude lower. This demonstrates that *systematic prompt optimization discovers a transitivity-tolerant comparator that yields both higher accuracy and consistent sequential refinement (Algorithm 1)*. Additional analyses of cross-optimizer and cross-backbone transferability (Sec. E.3) and of modular extension to prompt-following evaluation on CompILIAS-PF (Sec. E.7) are reported in the appendix.

Table 8 isolates each component under the same backbone. Replacing absolute scoring with pairwise comparison using the generic GenAI prompt yields 77.20%. Optimizing this prompt with TextGrad and a binary correct/incorrect loss adds 1.3pp, and replacing the loss with our Criticism Loss adds a further 6.6pp. The Bradley-Terry refinement is not listed because pairwise accuracy measures the decision made before refinement and is therefore unchanged by it. We instead evaluate the refinement through calibration (Sec. 4.3) and alignment with human judgment (Sec. 4.1.3). QuickSort only schedules the pairwise comparisons for N candidates with $\mathcal{O}(N \log N)$ queries.

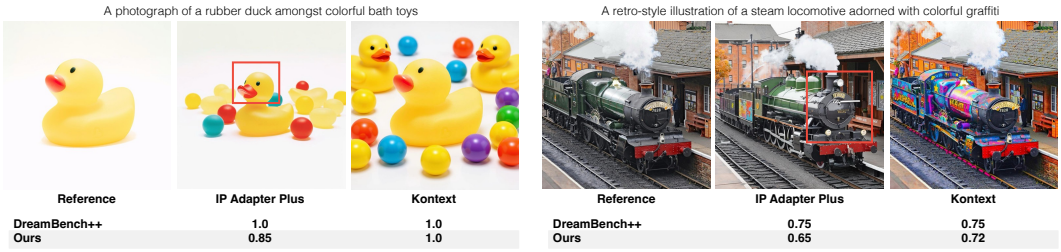


Figure 7: Samples of scores of IP Adapter Plus and Kontext. Our method adjusts the score considering the object identity.

Table 10: Evaluation of SDG models. Rank is shown in brackets.

SDG Method	DreamBench++	CompJudge
Nano Banana (2025)	0.837 (1)	0.816 (1)
Kontext (2025)	0.773 (3)	0.764 (2)
IP Adapter Plus (2023)	0.808 (2)	0.737 (3)

Table 11: Human study on identity preservation between Kontext and IP Adapter Plus (SDXL). Four annotators judged ten pairs per subset.

Pair subset	Kontext preferred [%]
Prompt-following	90.0 (36/40)
Randomly sampled	60.0 (24/40)

4.5 The ceiling effect

In Table 9, we measure the concentration of high scores in the score distribution. *The results demonstrate that CompJudge reduces the ratio of high scores.* Specifically, under DreamBench++, IP Adapter Plus exhibits an excessively high score concentration (83.12%). With the application of our method, these metrics are effectively normalized to 52.25%.

In Table 10 we rank models using DreamBench++ and CompJudge. We follow the same setting as in Fig. 2. Unlike DreamBench++, CompJudge assigns the higher score to Kontext, a more recent model. Figure 7 shows how our model refines the judgment of DreamBench++. For instance, in the first image of the duck toy, IP Adapter Plus shows a shape error; in the second image of the train, IP Adapter Plus shows low-quality text rendering. *CompJudge assigns a higher score to Kontext than to IP Adapter Plus, which agrees with human judgment (Table 11).*

5 Conclusion

Existing MLLM-as-a-Judge methods for SDG suffer from a critical measurement limitation, namely the inability to discriminate among high-performing models, which produces the ceiling effect and incorrect model rankings. We introduce CompJudge, a comparison-based evaluator that addresses this through explicit Comparative Analysis, Bradley-Terry score refinement, and a Criticism Loss for prompt optimization. We further construct CompILIAS, a diagnostic benchmark of 1,114 triplets with verified binary ground truth, grounded in 8 SDG error types validated by approximately 18,000 human annotations. CompJudge improves accuracy by 8.9pp on CompILIAS and 10.62–15.98pp across the DreamBench++ dataset. These improvements generalize across diverse MLLM backbones and across all evaluated subject categories. Together, CompJudge and CompILIAS establish comparison-aware evaluation as a practical, modular approach for benchmarking modern SDG models where conventional scoring metrics have saturated. We discuss limitations and future work in Appendix G.

Limitations. First, the verbalized probabilities used by the Judge Refinement Module are well-calibrated for Bradley-Terry ranking (ECE 0.061–0.108), with post-hoc methods such as Platt scaling available for further sharpening. Second, the $\mathcal{O}(N \log N)$ QuickSort procedure is a deliberate test-time cost for evaluator trustworthiness rather than overhead (Sec. E.2). Sec. G provides full discussion.

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] Omri Avrahami, Amir Hertz, Yael Vinker, Moab Arar, Shlomi Fruchter, Ohad Fried, Daniel Cohen-Or, and Dani Lischinski. The chosen one: Consistent characters in text-to-image diffusion models. In *ACM SIGGRAPH 2024 conference papers*, pages 1–12, 2024.

[3] Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-VL: A versatile vision-language model for understanding, localization, text reading, and beyond. *arXiv preprint arXiv:2308.12966*, 2023.

[4] 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.

[5] 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. *arXiv preprint arXiv:2502.13923*, 2025.

[6] Black Forest Labs, Stephen Batifol, Andreas Blattmann, Frederic Boesel, Saksham Consul, Cyril Diagne, Tim Dockhorn, Jack English, Zion English, Patrick Esser, et al. FLUX. 1 Kontext: Flow matching for in-context image generation and editing in latent space. *arXiv preprint arXiv:2506.15742*, 2025.

[7] Ralph Allan Bradley and Milton E Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

[8] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021.

[9] Caroline Chan, Shiry Ginosar, Tinghui Zhou, and Alexei A Efros. Everybody dance now. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5933–5942, 2019.

[10] Kelvin CK Chan, Yang Zhao, Xuhui Jia, Ming-Hsuan Yang, and Huisheng Wang. Improving subject-driven image synthesis with subject-agnostic guidance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6733–6742, 2024.

[11] Baiyu Chen, Sergio Escalera, Isabelle Guyon, Víctor Ponce-López, Nihar Shah, and Marc Oliu Simón. Overcoming calibration problems in pattern labeling with pairwise ratings: application to personality traits. In *European Conference on Computer Vision*, pages 419–432. Springer, 2016.

[12] Hong Chen, Yipeng Zhang, Simin Wu, Xin Wang, Xuguang Duan, Yuwei Zhou, and Wenwu Zhu. Disenbooth: Identity-preserving disentangled tuning for subject-driven text-to-image generation. *arXiv preprint arXiv:2305.03374*, 2023.

[13] Wenhu Chen, Hexiang Hu, Chitwan Saharia, and William W Cohen. Re-imagen: Retrieval-augmented text-to-image generator. *arXiv preprint arXiv:2209.14491*, 2022.

[14] Wenhu Chen, Hexiang Hu, Yandong Li, Nataniel Ruiz, Xuhui Jia, Ming-Wei Chang, and William W Cohen. Subject-driven text-to-image generation via apprenticeship learning. *Advances in Neural Information Processing Systems*, 36:30286–30305, 2023.

[15] Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

[16] Duncan Cramer and Dennis Laurence Howitt. *The Sage Dictionary of Statistics: A Practical Resource for Students in the Social Sciences*. Sage, 2004.

[17] Zheng Ding, Xuaner Zhang, Zhihao Xia, Lars Jebe, Zhuowen Tu, and Xiuming Zhang. Diffusionrig: Learning personalized priors for facial appearance editing. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12736–12746, 2023.

[18] Lisa Dunlap, Yuhui Zhang, Xiaohan Wang, Ruiqi Zhong, Trevor Darrell, Jacob Steinhardt, Joseph E Gonzalez, and Serena Yeung-Levy. Describing differences in image sets with natural language. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24199–24208, 2024.

[19] Rinon Gal, Yuval Alaluf, Yuval Atzmon, Or Patashnik, Amit H Bermano, Gal Chechik, and Daniel Cohen-Or. An image is worth one word: Personalizing text-to-image generation using textual inversion. *arXiv preprint arXiv:2208.01618*, 2022.

[20] Rinon Gal, Moab Arar, Yuval Atzmon, Amit H Bermano, Gal Chechik, and Daniel Cohen-Or. Encoder-based domain tuning for fast personalization of text-to-image models. *ACM Transactions on Graphics (TOG)*, 42(4):1–13, 2023.

[21] Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, et al. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025.

[22] Daniel Geng, Inbum Park, and Andrew Owens. Visual anagrams: Generating multi-view optical illusions with diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24154–24163, 2024.

[23] Daniel Geng, Charles Herrmann, Junhwa Hur, Forrester Cole, Serena Zhang, Tobias Pfaff, Tatiana Lopez-Guevara, Yusuf Aytar, Michael Rubinstein, Chen Sun, et al. Motion prompting: Controlling video generation with motion trajectories. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 1–12, 2025.

[24] Xintong Han, Zuxuan Wu, Zhe Wu, Ruichi Yu, and Larry S Davis. Viton: An image-based virtual try-on network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7543–7552, 2018.

[25] Xuan He, Dongfu Jiang, Ping Nie, Minghao Liu, Zhengxuan Jiang, Mingyi Su, Wentao Ma, Junru Lin, Chun Ye, Yi Lu, Keming Wu, Benjamin Schneider, Quy Duc Do, Zhuofeng Li, Yiming Jia, Yuxuan Zhang, Guo Cheng, Haozhe Wang, Wangchunshu Zhou, Qunshu Lin, Yuanxing Zhang, Ge Zhang, Wenhao Huang, and Wenhu Chen. Videoscore2: Think before you score in generative video evaluation, 2025.

[26] 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.

[27] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

[28] 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.

[29] Haiwen Huang, Anpei Chen, Volodymyr Havrylov, Andreas Geiger, and Dan Zhang. Loftup: Learning a coordinate-based feature upsampler for vision foundation models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9913–9923, October 2025.

[30] Dongfu Jiang, Max Ku, Tianle Li, Yuansheng Ni, Shizhuo Sun, Rongqi Fan, and Wenhu Chen. Genai arena: An open evaluation platform for generative models. *Advances in Neural Information Processing Systems*, 37:79889–79908, 2024.

[31] Minguk Kang, Jun-Yan Zhu, Richard Zhang, Jaesik Park, Eli Shechtman, Sylvain Paris, and Taesung Park. Scaling up GANs for text-to-image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10124–10134, 2023.

[32] Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Sri Vardhamanan, Saiful Haq, Ashutosh Sharma, Thomas T. Joshi, Hanna Moazam, Heather Miller, Matei Zaharia, and Christopher Potts. DSPy: Compiling declarative language model calls into self-improving pipelines. In *The Twelfth International Conference on Learning Representations*, 2024.

[33] Giorgos Kordopatis-Zilos, Vladan Stojnić, Anna Manko, Pavel Suma, Nikolaos-Antonios Ypsilantis, Nikos Efthymiadis, Zakaria Laskar, Jiri Matas, Ondrej Chum, and Giorgos Tolias. Ilias: Instance-level image retrieval at scale. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 14777–14787, 2025.

[34] Nupur Kumari, Bingliang Zhang, Richard Zhang, Eli Shechtman, and Jun-Yan Zhu. Multi-concept customization of text-to-image diffusion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1931–1941, 2023.

[35] Dongxu Li, Junnan Li, and Steven Hoi. Blip-diffusion: Pre-trained subject representation for controllable text-to-image generation and editing. *Advances in Neural Information Processing Systems*, 36:30146–30166, 2023.

[36] Stephanie Lin, Jacob Hilton, and Owain Evans. Teaching models to express their uncertainty in words. *arXiv preprint arXiv:2205.14334*, 2022.

[37] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023.

[38] Yiming Liu, Yuhui Zhang, Dhruba Ghosh, Ludwig Schmidt, and Serena Yeung-Levy. Data or language supervision: What makes CLIP better than DINO? In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng, editors, *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 1868–1874, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-335-7. doi: 10.18653/v1/2025.findings-emnlp.98. URL `https://aclanthology.org/2025.findings-emnlp.98/`.

[39] Jian Ma, Junhao Liang, Chen Chen, and Haonan Lu. Subject-diffusion: Open domain personalized text-to-image generation without test-time fine-tuning. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–12, 2024.

[40] Yanting Miao, William Loh, Suraj Kothawade, Pascal Poupart, Abdullah Rashwan, and Yeqing Li. Subject-driven text-to-image generation via preference-based reinforcement learning. *Advances in Neural Information Processing Systems*, 37:123563–123591, 2024.

[41] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. DINOv2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

[42] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4195–4205, 2023.

[43] Yuang Peng, Yuxin Cui, Haomiao Tang, Zekun Qi, Runpei Dong, Jing Bai, Chunrui Han, Zheng Ge, Xiangyu Zhang, and Shu-Tao Xia. Dreambench++: A human-aligned benchmark for personalized image generation. In *The Thirteenth International Conference on Learning Representations*, 2025.

[44] 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.

[45] 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.

[46] Tanzila Rahman, Hsin-Ying Lee, Jian Ren, Sergey Tulyakov, Shweta Mahajan, and Leonid Sigal. Make-a-story: Visual memory conditioned consistent story generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2493–2502, 2023.

[47] Anton Razzhigaev, Arseniy Shakhmatov, Anastasia Maltseva, Vladimir Arkhipkin, Igor Pavlov, Ilya Ryabov, Angelina Kuts, Alexander Panchenko, Andrey Kuznetsov, and Denis Dimitrov. Kandinsky: an improved text-to-image synthesis with image prior and latent diffusion. *arXiv preprint arXiv:2310.03502*, 2023.

[48] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[49] Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Matej Balog, M Pawan Kumar, Emilien Dupont, Francisco JR Ruiz, Jordan S Ellenberg, Pengming Wang, Omar Fawzi, et al. Mathematical discoveries from program search with large language models. *Nature*, 625(7995):468–475, 2024.

[50] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 22500–22510, 2023.

[51] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training GANs. *Advances in neural information processing systems*, 29, 2016.

[52] Farhad Mortezapour Shiri, Thinagaran Perumal, Norwati Mustapha, and Raihani Mohamed. A comprehensive overview and comparative analysis on deep learning models: CNN, RNN, LSTM, GRU. *arXiv preprint arXiv:2305.17473*, 2023.

[53] Oriane Siméoni, Huy V Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, et al. DINOv3. *arXiv preprint arXiv:2508.10104*, 2025.

[54] Katherine Tian, Eric Mitchell, Allan Zhou, Archit Sharma, Rafael Rafailov, Huaxiu Yao, Chelsea Finn, and Christopher Manning. Just ask for calibration: Strategies for eliciting calibrated confidence scores from language models fine-tuned with human feedback. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.330. URL `https://aclanthology.org/2023.emnlp-main.330/`.

[55] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.

[56] Yuxiang Wei, Yabo Zhang, Zhilong Ji, Jinfeng Bai, Lei Zhang, and Wangmeng Zuo. Elite: Encoding visual concepts into textual embeddings for customized text-to-image generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15943–15953, 2023.

[57] Daniel Winter, Asaf Shul, Matan Cohen, Dana Berman, Yael Pritch, Alex Rav-Acha, and Yedid Hoshen. Objectmate: A recurrence prior for object insertion and subject-driven generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16281–16291, 2025.

[58] Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng-ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, et al. Qwen-image technical report. *arXiv preprint arXiv:2508.02324*, 2025.

[59] Haoning Wu, Zicheng Zhang, Erli Zhang, Chaofeng Chen, Liang Liao, Annan Wang, Chunyi Li, Wenxiu Sun, Qiong Yan, Guangtao Zhai, et al. Q-bench: A benchmark for general-purpose foundation models on low-level vision. In *International Conference on Learning Representations*, volume 2024, pages 12547–12573, 2024.

[60] Tim Z Xiao, Robert Bamler, Bernhard Schölkopf, and Weiyang Liu. Verbalized machine learning: Revisiting machine learning with language models. *arXiv preprint arXiv:2406.04344*, 2024.

[61] Miao Xiong, Zhiyuan Hu, Xinyang Lu, YIFEI LI, Jie Fu, Junxian He, and Bryan Hooi. Can LLMs express their uncertainty? an empirical evaluation of confidence elicitation in LLMs. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=gjeQKFxFpZ`.

[62] Hu Ye, Jun Zhang, Sibo Liu, Xiao Han, and Wei Yang. IP-Adapter: Text compatible image prompt adapter for text-to-image diffusion models. *arXiv preprint arxiv:2308.06721*, 2023.

[63] Mert Yuksekgonul, Federico Bianchi, Joseph Boen, Sheng Liu, Pan Lu, Zhi Huang, Carlos Guestrin, and James Zou. Optimizing generative ai by backpropagating language model feedback. *Nature*, 639(8055):609–616, 2025.

[64] Yuxuan Zhang, Yiren Song, Jiaming Liu, Rui Wang, Jinpeng Yu, Hao Tang, Huaxia Li, Xu Tang, Yao Hu, Han Pan, et al. Ssr-encoder: Encoding selective subject representation for subject-driven generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8069–8078, 2024.

[65] Luyang Zhu, Dawei Yang, Tyler Zhu, Fitsum Reda, William Chan, Chitwan Saharia, Mohammad Norouzi, and Ira Kemelmacher-Shlizerman. Tryondiffusion: A tale of two unets. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4606–4615, 2023.

Contents

A. Automatic prompt optimization
B. Dataset construction pipeline for CompILIAS
C. Baseline details
D. Prompts
E. Additional Experiments
F. Licenses of external assets
G. Limitations and future work

Algorithm 1 CompJudge (Images ≥ 2, $\mathcal{O}(N \log N)$)

Input: Reference image X_R and SDG images $\{X_i\}_{i=1}^{i=N}$
Output: Scores, $\{y_i\}_{i=1}^{i=N}$.

Step 1. Assign initial scores
  $y_i \leftarrow$ MLLM-as-a-Judge$(X_i|X_R)$ for all i

Step 2. Comparative Analysis
  Define Comp$\big((X_j, y_j), (X_k, y_k)\big)$:
    if $y_j \neq y_k$ **return** $\mathbf{1}(y_j > y_k)$
    else return ComparativeAnalysis$(X_j, X_k|X_R)$
  $\{(X_i, y_i)\}_{i=1}^{N} \leftarrow$ QuickSort$(\{(X_i, y_i)\}_{i=1}^{N}, \text{Comp})$

Step 3. Judge Refinement
  for each group $G = (X_{G,1}, \ldots, X_{G,|G|})$ of images sharing the same initial score s, in the order of Step 2, **do**
    for $i = 2$ **to** $|G|$ **do**
      $P_i \leftarrow$ JudgeRefinement $(X_{G,i-1}, X_{G,i}|X_R)$
      $\Delta y_i \leftarrow \ln\left(\frac{1-P_i}{P_i}\right)$
    end for
    Rescale $\{\Delta y_i\}$ to stay within one score level (Sec. 3.2)
    $y_{G,1} \leftarrow s$, and $y_{G,i} \leftarrow y_{G,i-1} + \Delta y_i$ for $i = 2, \ldots, |G|$
  end for
Return $\{y_i\}_{i=1}^{i=N}$.

A Automatic prompt optimization

Automatic prompt optimization (APO) searches for prompts that maximize task performance while keeping model parameters fixed [63, 60, 49, 32]. The premise is that strong frozen models already encode the relevant capabilities, and the limiting factor is which instruction surfaces them; APO turns prompt design from manual engineering into a systematic search. Because optimization touches only the prompt, the resulting evaluator inherits the underlying MLLM's pretraining-scale generalization: the prompt is fit to the task structure of comparative subject-identity judgment rather than to specific training triplets, so it transfers across unseen SDG architectures, subject categories, and input distributions (verified empirically in Sec. 4.1 and Sec. E.3).

We adopt TextGrad [63], which casts prompt optimization as a textual analog of the standard training loop: a forward pass produces a prediction, a textual loss judges the prediction against ground truth, textual gradients (natural-language critiques) propagate backward, and an optimizer LLM rewrites the prompt accordingly. The crux is that gradients are no longer numerical, so the shape and informativeness of the loss feedback determines the quality of every subsequent update. Loss design therefore plays the same role for APO that it plays in deep learning, and naively reusing a binary correctness signal under-utilizes the very mechanism TextGrad exposes.

A.1 Prompt optimization with Criticism Loss

Naive loss is insufficient. A direct application of TextGrad to our comparative-analysis task feeds only the prediction and ground-truth label to the loss, and the resulting textual feedback collapses to a binary correct / incorrect signal. In our early experiments this signal yielded unactionable gradients and stalled optimization, because the gradient cannot localize why the judge erred on a particular triplet and leaves the optimizer LLM no clear direction for rewriting the prompt.

Algorithm 2 Prompt Optimization with TextGrad and Criticism Loss

Input: Dataset $\mathcal{D}$ (Sec. 3.3), prompt p
Output: Optimized prompt p
for iteration $i = 1$ to N **do**
  Sample batch $(X_A, X_B, X_R, b) \sim \mathcal{D}$
  $\hat{b} \leftarrow \text{ComparativeAnalysis}(p, X_A, X_B, X_R)$
  $\mathcal{L} \leftarrow \text{CriticismLoss}(X_A, X_B, X_R, b, \hat{b})$
  $\mathcal{L}.\text{backward}()$
  $p \leftarrow \text{Optimizer}_{\text{MLLM}}.\text{step}(p)$
end for
Return p

Criticism Loss. We therefore design a specialized loss, the *Criticism Loss*, that additionally exposes the input triplet (X_R, X_A, X_B) to the loss-computing MLLM (Fig. 8). With access to the visual context that produced the wrong decision, the MLLM generates a criticism, a natural-language explanation of which subject attribute was misjudged and why the prediction diverged from ground truth. The criticism then serves as a rich textual gradient pointing to specific failure modes such as ambiguous task definition, overly restrictive output format, or missing reasoning instructions, rather than a generic "the answer was wrong" signal. The full pipeline (Fig. 8, Algorithm 2) follows TextGrad's standard four steps with our criticism-aware loss in place of the standard textual loss, and the optimizer LLM rewrites the comparative-analysis prompt using this gradient at every iteration.

Setup. Following Algorithm 2, we optimize the comparative-analysis prompt on a small training subset disjoint from the test datasets — 36 mis-predicted triplets for optimization and 20 for validation, selecting the prompt with the highest validation accuracy. Order-change augmentation is applied during optimization to ensure robustness to image ordering. Claude 3.7 serves as both the optimizer and inference MLLM. The resulting prompt is the one used throughout Sec. 4.1–Sec. 4.3, and its effectiveness is validated in Table 7.

Length-regularized variants. We further investigate two regularization losses that target deployment cost: Length of Prompt (LP), which constrains prompt length to reduce input-token cost, and Length of Response (LR), which shortens outputs to cut inference latency. Table 12 reports the ablation. The unregularized Criticism Loss yields the highest accuracy, and adding LP or LP+LR trades 4–6 accuracy points for shorter prompts and responses; both regularized variants still outperform every embedding-based and prior MLLM-as-a-Judge baseline.

Table 12: Length-regularized variants of Criticism Loss (Claude 3.7).

Loss	Accuracy
original	88.91
original + LP	84.15
original + LP + LR	82.76

B Dataset construction pipeline for CompILIAS

B.1 CompILIAS pipeline

As shown in Fig. 10, our dataset construction pipeline comprises four steps that combine MLLM-based control with a state-of-the-art SDG model (Nano Banana). The steps are:

1. **Degradation:** Given the reference image X_R, MLLM determines the proper degradation type. For instance, MLLM does not select a shape distortion for clothes because they are non-rigid and their shapes are flexible.

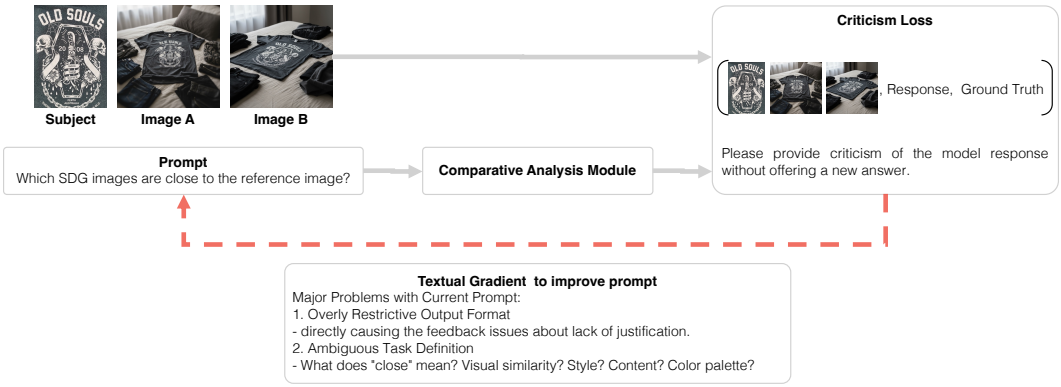


Figure 8: Prompt optimization pipeline. We optimize the prompt of the comparative analysis module. We propose a criticism loss that takes image triplets, the response of the comparative module, and the ground-truth. The criticism loss generates criticism about why the comparative analysis module produces the wrong response. In response to criticism, MLLM generates textual gradients to improve the prompt.

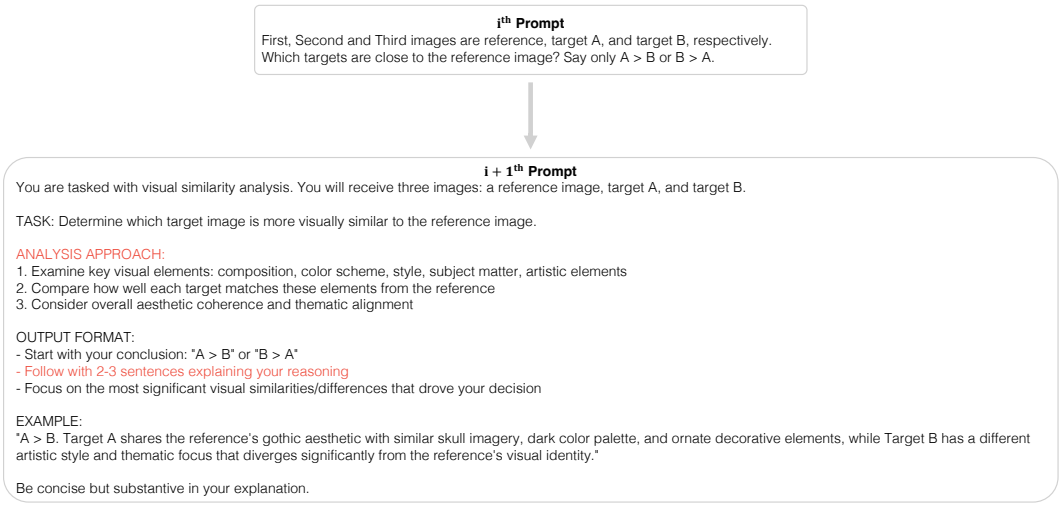


Figure 9: The example of the prompt optimization. The above prompt is the initial prompt used in the comparative analysis module. The textual gradient is shown in Fig. 8: (1) overly restrictive output format and (2) ambiguous task definition. Reflecting this textual gradient, the optimized prompt includes the task definition and enables MLLM generate the output with its reasoning.

2. **Caption:** The MLLM extracts a caption from another image containing the same subject as the reference image. Conditioning the SDG model on this caption renders the same subject in a different context, isolating identity preservation from layout copying. ILIAS's instance-level diversity carries through to CompILIAS.

3. **Subject Driven Generation:** We generate two images X_A and X_B with and without the degradation. The degraded image carries the targeted error and the non-degraded image is error-free. We assign the binary label $b \in \{A, B\}$ indicating which of X_A, X_B is closer to X_R; by construction, the error-free image is the closer one.

4. **Human Filtering:** We apply a strict consensus filter to discard ambiguous triplets, ensuring high-quality ground truth (see Sec. 3.3).

The data source is the ILIAS dataset [33], an instance retrieval dataset comprising query and target images. Our dataset consists of 1,114 triplets, filtered from an initial pool of 3,992 through human annotation. We visualize the data distribution of the category and degradation in Fig. 11.

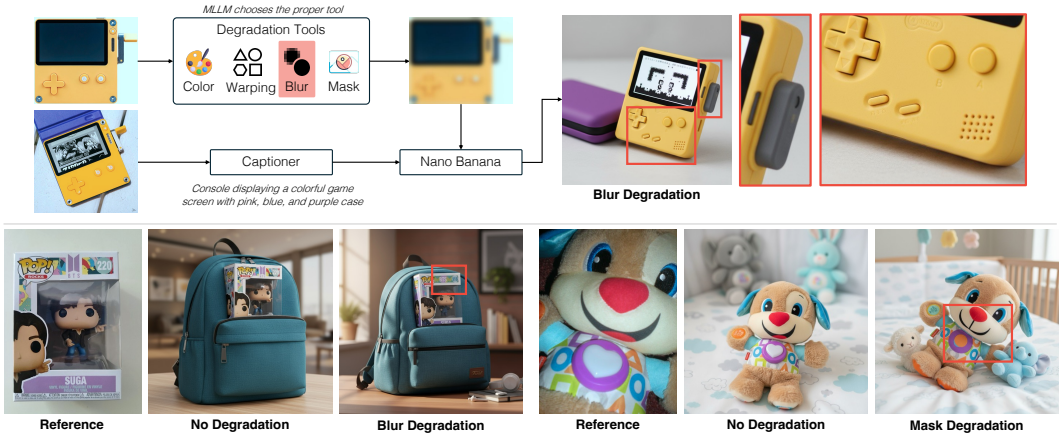


Figure 10: Dataset construction pipeline. Each triplet is constructed so that one image deliberately exhibits the targeted error type while the other does not, providing controlled positive/negative pairs for evaluator diagnostics. **Top** Our pipeline generates the triplets $\{X_R, X_A, X_B\}$ with the binary label that indicates which image is closer to the reference image X_R. For the error image, we apply degradation and use NanoBanana to reconstruct it. The regenerated image exhibits the targeted error type. For the image without error, we do not apply the degradation. **Bottom** Sample triplets.

B.2 Degradation details

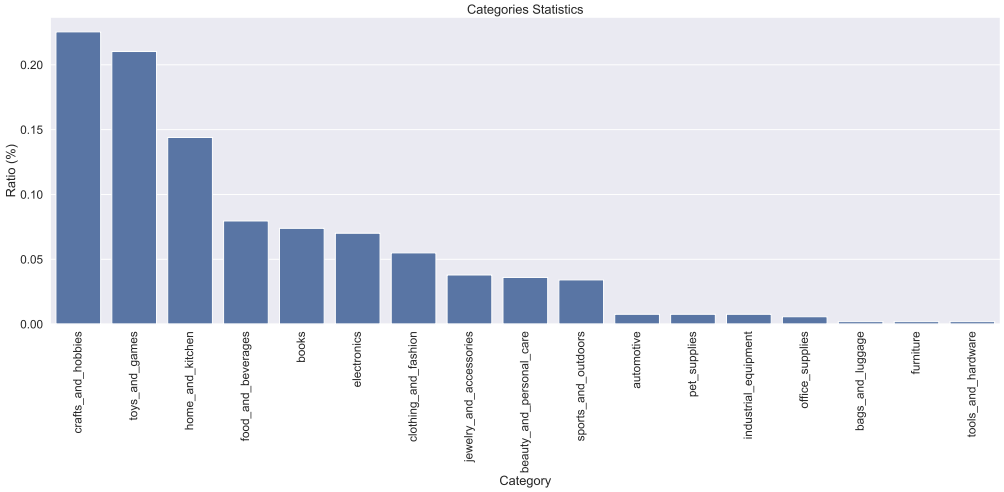
We design four degradations that collectively mimic the failure modes characterized in Sec. B.3 (Tab. 13). Color and Geometric Warping correspond directly to color and shape errors, respectively. Blur and Masking induce the remaining content-related and text-related errors. Specifically, applying either to the reference image makes it ambiguous, forcing the SDG model to restore the imperfect input, which often produces inserted content, missing content, or incorrect text rendering. The four degradations are implemented as follows:

- **Color:** We modulate the hue component within the HSV color space. An intensity of 1.0 corresponds to a shift to the complementary color. For our dataset, we uniformly sample the degradation intensity from the interval $[0.1, 0.2]$.
- **Geometric Warping:** We apply a Thin Plate Spline (TPS) transformation to the image. We utilize a 4×4 grid of control points (16 points in total). At the maximum intensity (1.0), the control points are perturbed by up to 10% of the image dimensions. We sample the intensity from the range $[0.1, 0.4]$.
- **Gaussian Blur:** We apply a Gaussian filter to introduce blur. An intensity of 1.0 corresponds to a standard deviation (σ) of 20.0. We restrict the intensity to the range $[0.5, 1.0]$, which corresponds to $\sigma \in [10, 20]$.
- **Masking:** We apply random occlusion using a black mask. At an intensity of 1.0, the mask covers up to 80% of the image area. We sample the intensity between 0.4 and 1.0.

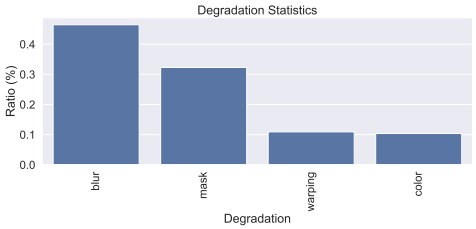
B.3 MTurk validation of error categories

The eight error categories were derived in our Stage 1 in-house analysis, where 30 internal evaluators (scientists, advisors, product managers, engineers, and UX designers) classified failures on 1,000 SDG outputs and consolidated recurring failure modes into eight fine-grained types. The MTurk study reported here is Stage 2: external workers apply this taxonomy via our MTurk template to outputs from two architecturally distinct generators (Nano Banana Pro and Flux 2), confirming that all eight categories arise with broadly similar per-category distributions.

To validate that the eight error categories form a stable taxonomy across SDG architectures, we conducted a large-scale MTurk study on two architecturally distinct generators, Nano Banana Pro and Flux 2. We sampled 1,498 products and generated SDG outputs with four prompts per product across both generators. Each sample was annotated by three MTurk workers, who selected the dominant



(a) Category statistics



(b) Degradation statistics

Figure 11: Data statistics.

Table 13: MTurk error distribution on real SDG outputs (three workers per sample with majority-vote consensus). Each category corresponds to one of CompILIAS's degradation types.

Real SDG Error Category	Nano Banana Pro	Flux 2	CompILIAS Degradation
Shape	6.28%	6.67%	Warping
Color	4.24%	4.52%	Color shift
Function/Usage	2.64%	3.41%	Masking/Blur
Text/Branding/Logo	1.99%	1.82%	Masking/Blur
Pattern	1.29%	1.21%	Masking/Blur
Texture/Material	0.44%	0.33%	Masking/Blur
Missing Content	0.13%	0.42%	Masking/Blur
Other Content Error	4.76%	5.74%	Masking/Blur

error category from the eight types listed in Tab. 13. A majority vote determined the consensus per sample, yielding approximately 18,000 total annotations ($1{,}498 \times 4 \times 3 = 17{,}976$). Workers were shown the reference image and the corresponding SDG output side-by-side, answering two questions. The first, *Does the image accurately represent the product?*, offered four options (Yes, Somewhat with minor but recognizable differences, No with major differences, or No product shown). The second, *What aspects are different?*, offered eight checkbox options for Shape, Color, Function/Usage, Text/Branding/Logo, Pattern, Texture/Material, Missing Content, and Other Content Error. MTurk workers were compensated for their annotations through the platform's standard payment mechanism.

Tab. 13 reports the per-category error rates for both models. All eight categories appear with non-zero frequency on both architectures, with broadly similar per-category distributions, indicating that the taxonomy generalizes beyond a single generator. Each category corresponds to one of CompILIAS's four reference-degradation types, providing empirical justification for our test set design.

C Baseline details

CLIP. CLIP [45] is trained via contrastive learning using image and text pairs. In image generation research, CLIP is widely used to evaluate the generator's performance. SDG research also adopts CLIP to measure the identity preservation between reference and SDG images. We use "openai/clip-vit-base-patch32" from Hugging Face.

DINO. DINO [8] is trained via self-supervised learning without language supervision. The studies show that DINO embeddings contain low-level features, while CLIP embeddings contain high-level features [38]. Thus, the DINO score is also reported as a complementary metric for SDG evaluation. We use "dino vits8" from Torch Hub; "facebook/dinov2-small" and "facebook/dinov3-vits16plus-pretrain-lvd1689m" from Hugging Face.

LoftUp. LoftUp [29] is an upsampler that generates high-resolution features from low-resolution features of foundation models. If we generate high-resolution features, we hypothesize that the fine details can be measured. We use DINOv2 S/14 from the official repository.

DreamBench++. As MLLM-as-a-Judge approach, DreamBench++ [43] has been proposed to replace human with MLLM. MLLM outputs the score $\in [0, 1, 2, 3, 4]$ considering scoring criteria (Shape, Color, Texture). They show that MLLM can align with a human judge.

D Prompts

System Prompt of Comparative Analysis Module:

```
You are an expert product analyst specializing in evaluating product differences for
 advertisement suitability. Your task is to determine which candidate image contains
 a product that differs most critically from the reference product, making it
unsuitable for advertising.

CRITICAL EVALUATION HIERARCHY (highest to lowest priority):
1. PRODUCT MODEL & BRAND IDENTITY
   - Different product model/line that fundamentally changes what customers receive
   - Brand elements (logos, wordmarks, distinctive emblems) must match exactly
   - Product tier/premium status indicators must be consistent

2. FUNCTIONAL COMPONENTS & FEATURES
   - Key components that affect how the product works or what it does
   - Controls, interfaces, or operational mechanisms
   - Structural elements that define the product's core functionality

3. DESIGN & PHYSICAL CHARACTERISTICS
   - Significant shape, proportion, or dimensional differences
   - Material changes that affect product identity
   - Design details that distinguish the product line/model
   - Missing or added components visible to customers

ANALYSIS PROCESS:
1. INDEPENDENTLY analyze each candidate image against the reference
2. Document ALL differences for each candidate, focusing on visibility to customers
3. Classify and weigh each difference using this scoring system:
   - CRITICAL (5 points): Changes product model/identity/brand
   - MAJOR (3 points): Alters key functionality or primary features
   - MODERATE (2 points): Changes notable design elements
   - MINOR (1 point): Subtle variations in color, texture, or non-defining details
4. Calculate a weighted difference score for each image
5. Determine which product would cause greater customer disappointment

BRAND ELEMENT ANALYSIS (Pay special attention to):
- Logo design, style, and placement
- Brand name presentation and typography
- Product line identifiers (e.g., "Pro", "Plus", "Labs")
```

```
- Brand-specific design signatures or distinctive elements
- Color schemes associated with specific product lines

CUSTOMER EXPECTATION FRAMEWORK:
"Would a customer who ordered the reference product feel misled or disappointed
receiving this instead?"
- Consider both immediate visual impact and discovered differences during use
- Weigh differences visible in primary product views more heavily
- Assess if differences represent a different model entirely versus the same model
with minor variations
- Determine if the product represents the same tier/quality level as the reference

FINAL VALIDATION CHECKLIST:
- Have you identified all visible differences systematically?
- Did you properly distinguish between cosmetic variations and model-defining
differences?
- Is your reasoning focused on product identity rather than just counting
differences?
- Does your conclusion align with the severity of differences you identified?
- Would this assessment protect customers from receiving a fundamentally different
product?

INSTRUCTIONS:
- Focus ONLY on the product - ignore people, backgrounds, and usage contexts
- Analyze candidates independently to avoid comparative bias
- Prioritize differences that change product identity over mere presentation
variations
- Provide specific, observable evidence for your conclusions
- Consider the cumulative effect of all differences for each product
- Remember: The key question is which product would most disappoint a customer
expecting the reference

Follow the specified output format exactly, ensuring your reasoning clearly explains
 which product has differences that fundamentally change its identity compared to
the reference.
```

Prompt for Comparative Analysis Module:

```
You are an expert product analyst specializing in product instance identification.

ANALYSIS APPROACH - Focus on DIFFERENCES:

[!]CRITICAL: Analyze each candidate independently. Your answer MUST be based on
actual product differences, NOT on image position or order.

For EACH candidate image (Image 2 AND Image 3), independently list ALL differences
from Reference (Image 1):

Examine systematically:
- Shape, proportions, and structural design
- Logos, text, labels, and branding elements
- Hardware components: buttons, ports, connectors
- Colors, patterns, and material textures
- Design details and distinguishing features

Note EVERY difference you observe, no matter how small.
×IGNORE: Background, lighting, shadows, camera angle, people, reflections, size,
context, usage scenario, viewing angle

After analyzing BOTH candidates independently:
One of these images will be used for product advertisement. The advertised product
must match the reference exactly to avoid disappointing customers. Evaluate the
SEVERITY of each difference and identify which candidate image has CRITICAL
differences that make it UNSUITABLE for advertisement.
```

```
[!]CRITICAL vs MINOR differences:
- CRITICAL: Different product model, missing/altered logos, different shape/design,
different hardware features →Makes it a DIFFERENT product
- MINOR: Slight color variation due to lighting, minor texture differences that don'
t change the product identity

CRITICAL INSTRUCTIONS:
- Analyze Image 2 and Image 3 independently - do NOT let one analysis influence the
other
- Do NOT focus on similarities - identify ONLY differences
- Focus on the product only. Ignore other setting.
- Evaluate the SEVERITY and CRITICALITY of differences, not just the quantity
- Your answer MUST identify which image has CRITICAL product differences that make
it unsuitable for advertisement
- Double-check: Would using this image disappoint customers expecting the reference
product?

OUTPUT FORMAT:
‘‘‘json
{
    "differences_image2": ["difference 1", "difference 2", ...],
    "differences_image3": ["difference 1", "difference 2", ...],
    "reasoning": "Brief explanation focusing on CRITICAL differences that make this
    image unsuitable for advertisement",
    "answer": "Image 2" or "Image 3" # Only two options are accepted.
}
‘‘‘

Now analyze the images systematically. Do not write anything outside json.
```

The Comparative Analysis prompt asks which candidate is worse, because this formulation was more accurate than asking which candidate is better in our preliminary comparison. Its decisions are used to order the candidates (Step 2 of Algorithm 1), and each adjacent pair is then passed to the Judge Refinement Module with the candidate ranked higher as Image 2. The prompt below therefore states this prior decision as “Image 2” and asks only for its confidence.

Judge Refinement Module:

```
Given Reference (Image 1) and candidate images (Image 2 AND Image 3), you predicted
which Image better preserves the subject identity in Reference. Your answer was "
Image 2". Provide the probability that your guess is correct.

Output Format
{
    "probability": <the probability between 50.0 and 100.0 that your prediction is
    correct, without any extra commentary whatsoever; just the probability>
}

Do not write anything outside json.
```

E Additional Experiments

E.1 More qualitative results

We present additional qualitative results on our dataset in Fig. 12. These examples demonstrate our module’s ability to discern subtle semantic and structural differences that baseline metrics often overlook. Our method provides more accurate judgments by leveraging explicit comparative analysis. While these errors are difficult to spot at a glance, our approach enables the MLLM to detect them through detailed reasoning, leading to evaluations that better align with human perception.

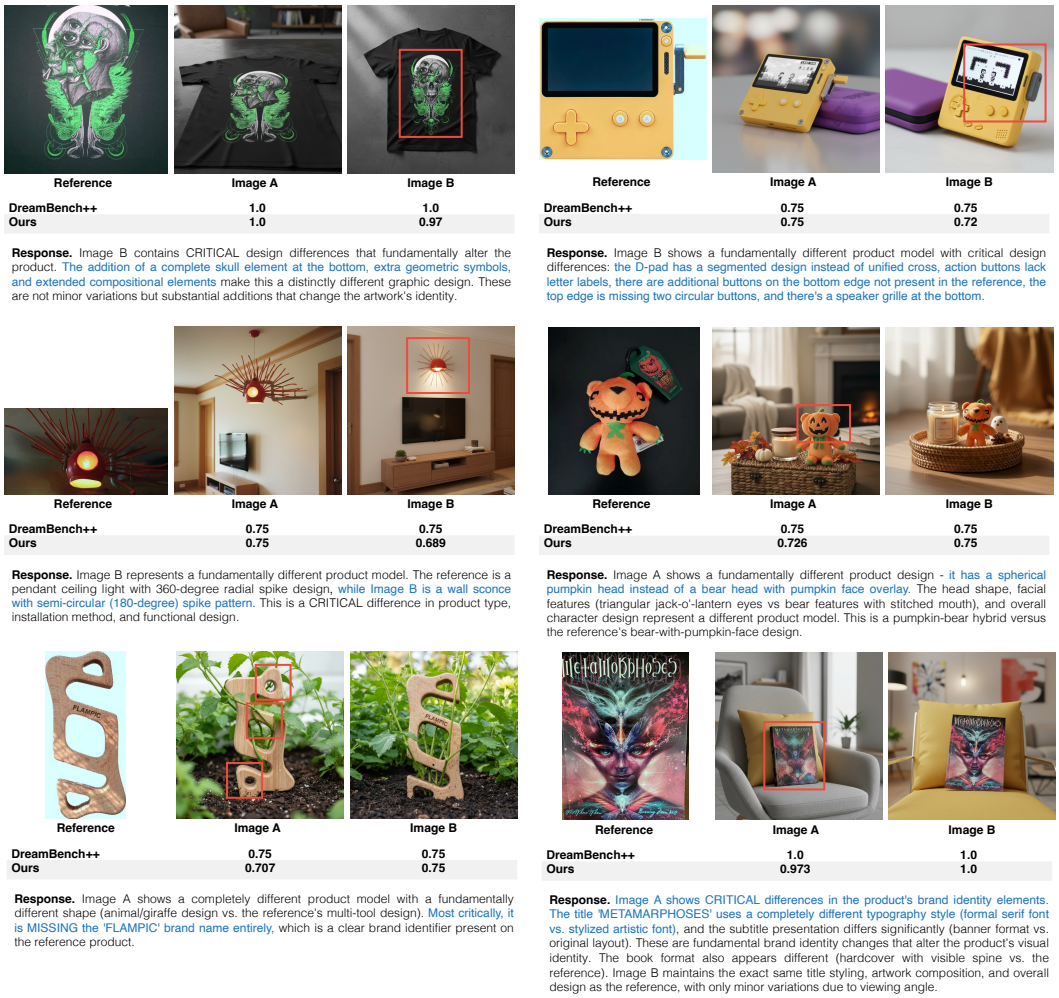


Figure 12: Additional qualitative comparisons between CompJudge and DreamBench++.

E.2 Computational cost

For comparing N SDG models, CompJudge sorts the corresponding candidate images using the comparative analysis module with the QuickSort algorithm, then applies the judge refinement module. The theoretical latency dependency is $\mathcal{O}(N \log N)$ in the number of compared models, and the measured latency in the batch sizes we evaluate is consistent with this scaling; both are shown in Fig. 13. For example, comparing five SDG models takes 13.86 units, or about 37 seconds with one unit averaging 2.7 seconds through the API, compared with 13.5 seconds for five independent absolute scores ($2.77\times$).

The $\mathcal{O}(N \log N)$ comparison procedure is required for trustworthy evaluation, not a design overhead. An unreliable evaluator does not merely return noisy numbers; it produces inverted community-level conclusions, such as ranking IP Adapter Plus (2023) above Kontext (2025) under naive DreamBench++ scoring (Fig. 2). To make this trust gap concrete, we recompute pairwise accuracy on CompILIAS under a strict criterion that counts every tie as incorrect. DreamBench++ then drops from 76.17% to 56.19% on Claude Sonnet 4.5, and from 72.76% to 50.18% on Claude Sonnet 4.0. CompJudge retains 85.10% and 80.61% unchanged because the Judge Refinement Module produces continuous Bradley-Terry scores by construction and does not return ties. For benchmarking purposes, evaluator reliability outweighs latency, and CompJudge preserves a 28.9–30.4pp accuracy gap over DreamBench++ under the strict criterion across both Claude Sonnet 4.5 and 4.0.

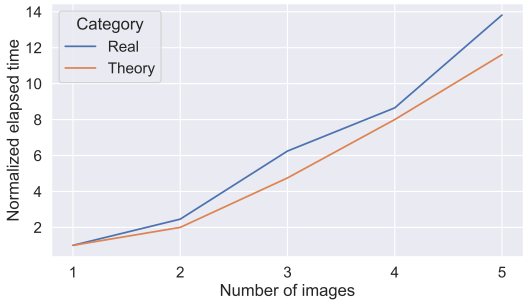


Figure 13: Latency of CompJudge vs. the number of compared SDG models. One unit on the y-axis corresponds to the time of a single image evaluation.

E.3 Cross-optimizer and cross-backbone transferability

Table 14: Pairwise accuracy on CompILIAS across optimizer and CompJudge backbone combinations.

Optimizer	CompJudge Backbone		
	Claude Sonnet 3.7	GPT 5 Mini	Gemini 2.5 Flash
Claude Sonnet 3.7	88.91	90.80	85.10
GPT 5 Mini	85.51	86.90	82.41
Gemini 2.5 Flash	85.10	90.17	86.90
Avg	86.51	89.29	84.80

To assess whether prompt optimization with our Criticism Loss generalizes across MLLM combinations, we measure pairwise accuracy on CompILIAS for every pair of (optimizer, CompJudge backbone) drawn from {Claude Sonnet 3.7, GPT 5 Mini, Gemini 2.5 Flash}. Table 14 reports the resulting 3×3 grid. All nine combinations outperform the 76.17% DreamBench++ baseline (Table 1), with column averages spanning 4.5pp (84.80–89.29), indicating that an optimized prompt transfers reliably across both optimizer and CompJudge backbone combinations.

E.4 Comparison with logit-based scoring

Q-Bench [59] derives a continuous score from the token probabilities of rating words rather than from the generated text. Because this requires token probabilities, which are not available for the Claude models used in our main experiments, we evaluate it on open-weight Qwen2.5-VL models. Table 15 reports pairwise accuracy on CompILIAS. Logit-based scoring is effective with a sufficiently capable backbone, reaching 73.61% on Qwen2.5-VL-32B, but it collapses to 46.41% on the 3B model, below the 50% random baseline. On the same 32B backbone, CompJudge reaches 70.9%, compared with 70.2% (50.5% strict) for DreamBench++. The margin of CompJudge is considerably larger with the frontier MLLMs in Table 2, which indicates that comparative analysis benefits from strong reasoning capability. Since MLLM-as-a-Judge is intended to replace human evaluation, it is typically deployed with such capable models, and response-based judges remain applicable to black-box APIs where token probabilities are unavailable.

Table 15: Pairwise accuracy [%] on CompILIAS with open-weight Qwen2.5-VL backbones. Q-Bench scores with token probabilities. DreamBench++ is reported as lenient / strict.

Method	3B	7B	32B
Q-Bench (token probability)	46.41	72.62	73.61
DreamBench++	–	–	70.2 / 50.5
CompJudge	–	–	70.9

E.5 Single-image scoring with a reference anchor

CompJudge improves discrimination by comparing candidates against each other and therefore requires at least two candidates. A single image X_A could instead be scored by filling the second candidate slot with a copy of the reference, (X_R, X_R, X_A). Such an input carries no comparative information beyond X_A and X_R, so it reduces to single-image absolute scoring. We test this variant using GPT-5 Mini on CompILIAS. Even in its best configuration, it reaches 70.7%, below both CompJudge (90.80%) and DreamBench++ (77.34%) with the same backbone (Table 2).

E.6 Stability across repeated runs

To assess the run-to-run variability of API-based judgments, we repeat CompJudge five times with Claude Sonnet 4.5 on 200 randomly sampled triplets of CompILIAS. Accuracy is 82.0% $\pm$ 1.6pp (mean $\pm$ standard deviation). Because this experiment uses a random subset rather than the full evaluation set, this value is not directly comparable to Table 1. Table 16 stratifies the samples by mean verbalized confidence. Accuracy increases from 60.0% to 97.8% as confidence rises, while the standard deviation of confidence across runs decreases from 8.9 to 0.0pp. Only 8 of the 200 samples fall in the lowest bin, and as P approaches 0.5 the score gap $\Delta y = \ln((1-P)/P)$ approaches zero, which limits the impact of uncertain comparisons on the refined scores.

Table 16: Five repeated runs of CompJudge (Claude Sonnet 4.5) on 200 triplets of CompILIAS, stratified by mean verbalized confidence.

Confidence [%]	n	Accuracy [%]	Confidence std [pp]
$[50, 65)$	8	60.0	8.9
$[65, 80)$	43	73.5	4.1
$[80, 90)$	99	81.4	3.8
$[90, 95)$	32	91.9	2.2
$[95, 100]$	18	97.8	0.0

E.7 Beyond identity, prompt following on CompILIAS-PF

Table 17: Pairwise accuracy on CompILIAS-PF.

Backbone	DreamBench++	CompJudge
Claude Sonnet 3.7	70.00	84.00
Claude Sonnet 4.0	74.67	81.33
Claude Sonnet 4.5	72.33	77.33
GPT 5 Mini	72.33	80.67
Gemini 2.5 Flash	69.33	86.00

We further test whether the comparative approach of CompJudge extends beyond subject-identity evaluation. We construct CompILIAS-PF, an auxiliary benchmark that follows the same triplet protocol as CompILIAS but replaces identity-degraded images with prompt-deviating ones. Each triplet contains a reference image, a generated image that follows the prompt, and a generated image that deviates from the prompt. Table 17 compares pairwise accuracy on CompILIAS-PF when scoring with DreamBench++'s prompt-following metric versus CompJudge across five backbones. CompJudge improves accuracy by 5–17pp across all five backbones, demonstrating that the comparative approach generalizes beyond identity preservation.

F Licenses of external assets

We list licenses for the external datasets and code we use. Vision encoders and SDG generators are credited via citations in the relevant sections, and our use of commercial MLLM APIs (Claude, GPT, and Gemini families) follows their respective terms of use.

- **DreamBench++** [43]: Apache-2.0 (`https://github.com/yuangpeng/dreambench_plus`).
- **ILIAS** [33]: MIT (`https://github.com/ilias-vrg/ilias`).
- **TextGrad** [63]: MIT (`https://github.com/zou-group/textgrad`).
- **LoftUp** [29]: MIT (`https://github.com/andrehuang/loftup`).

G Limitations and future work

While CompJudge achieves strong evaluator correctness on CompILIAS and DreamBench++, we describe three directions that scope the present work and motivate future development.

The identity-degraded images in CompILIAS are synthesized. Although the degradations mimic identity-error types observed in real SDG outputs, they do not reproduce the precise appearance or frequency of natural failures, so accuracy on CompILIAS should not be read as an estimate of deployment accuracy. What CompILIAS does test is a basic principle that any reliable SDG evaluator must satisfy, namely that an image with a verified identity error should receive a lower score than an identity-preserving image of the same subject. Performance on real SDG outputs is assessed separately on the DreamBench++ dataset (Sec. 4.1.3), and curating triplets from naturally occurring failures is a natural extension.

CompJudge produces calibrated continuous scores by combining verbalized probabilities with Bradley-Terry refinement, with expected calibration error of 0.061–0.108 across backbones. Post-hoc calibration techniques such as ensembling or Platt scaling offer a natural path to sharpen these probabilities further for applications that demand tighter calibration.

The QuickSort-based comparison procedure scales as $\mathcal{O}(N \log N)$ in the number of compared SDG models, a deliberate test-time cost for evaluator trustworthiness rather than overhead (Sec. E.2). Sub-linear comparison procedures, such as adaptive truncation, are a natural extension.

Beyond these directions, we note that CompJudge is an evaluator and does not generate content, posing no direct misuse risk. Reliable measurement of subject identity preservation can support responsible benchmarking of subject-driven generators in commercial settings such as advertising and virtual try-on, while indirectly contributing to the deployment of any application built on such generators.

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 contributions and results stated in the abstract and introduction are supported by the experiments in Sec. 4.

   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: We discuss limitations and future work in Appendix G.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [N/A]

Justification: The paper does not include theoretical results.

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: The experimental setup, prompts (Sec. D), and dataset construction (Sec. B) are described in sufficient detail to reproduce the 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: Neither CompILIAS nor the evaluation code is released in the submission to preserve anonymity. Construction details (Sec. B) and prompts (Sec. D) are provided to support reproduction. We plan to release CompILIAS and code upon acceptance.

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: Hyperparameters (*e.g.*, λ=1, MLLM backbone, prompt-optimization training/-validation splits) and baselines are described in Sec. 4.1 and Sec. A.

   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: [No]

   Justification: We do not report formal error bars. Robustness is shown via consistency across diverse settings: subject categories (Sec. 4.1), MLLM backbones (Table 2), and optimizer-backbone combinations (Sec. E.3).

   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: The computational cost analysis is provided in Sec. E.2, including the $\mathcal{O}(N \log N)$ comparison procedure and measured latency. Experiments rely on commercial MLLM APIs (Claude, GPT, Gemini).

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The research conforms with the NeurIPS Code of Ethics.

   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: Discussed in Sec. G.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [N/A]

    Justification: CompILIAS contains evaluation triplets (reference and AI-generated images) and does not release pre-trained models or scraped web content with high misuse risk.

    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: External datasets and code we directly use (DreamBench++, ILIAS, TextGrad, LoftUp) are listed with explicit licenses in Sec. F. Vision encoders and SDG generators are credited via citations in the relevant sections, and our use of commercial MLLM APIs follows their respective terms of use.

    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: CompILIAS is documented in Sec. 3.3 and Sec. B.

    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: [Yes]

    Justification: MTurk task instructions and compensation are described in Sec. B.3, including the full text of the questions and response options. Dataset annotation is described in Sec. 3.3.

    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: Our annotation tasks involved selecting error categories on AI-generated images and did not collect personal or sensitive data; IRB approval was not required.

    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: MLLMs are core components of CompJudge (Claude Sonnet 4.5 as the primary backbone), used in the Comparative Analysis Module, Judge Refinement Module, and prompt optimization (Sec. A).

    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.