CyCLeGen: Cycle-Consistent Layout Prediction and Image Generation

Xiaojun Shan,* **Haoyu Shen,*†** **Yucheng Mao, Haiyang Xu,**
Xiang Zhang, Abhay Anand, Bingnan Li, Zhuowen Tu
UC San Diego

Abstract

We present CyCLeGen, an autoregressive framework that integrates layout understanding and layout-to-image generation through *cycle consistency*: predicted layouts must produce faithful images, and generated images must yield consistent layouts. We enforce this constraint via **CycleGRPO**, a bidirectional reinforcement learning strategy with complementary geometric and perceptual rewards, enabling self-introspective learning from only 8k RL samples. This creates a natural loop, generation helps understanding by rewarding only those layouts that lead to high-quality images, and understanding helps generation by rewarding only those images whose spatial structure can be faithfully recovered. Extensive experiments show that CyCLeGen achieves significant gains across diverse image understanding and generation benchmarks, with emergent gains on image captioning.

1 Introduction

Layout understanding (image $\rightarrow$ layout) and layout-to-image generation (layout $\rightarrow$ image) are two structurally inverse tasks that together form the backbone of controllable generation: the former extracts spatial structure from images, while the latter synthesizes images conditioned on spatial structure. Recent autoregressive vision-language models [1, 2] have shown that both tasks can be handled within a single architecture, raising a natural question: *Can these two inverse tasks mutually reinforce each other?*

In practice, existing methods treat these two directions as loosely coupled objectives that share parameters but lack mutual constraints [1]. We identify a concrete failure mode: fine-tuning for layout understanding degrades generation fidelity, while fine-tuning for generation severely weakens understanding. This interference arises because neither direction receives feedback about whether its output is *recoverable* by the other — the understanding branch can predict a geometrically accurate layout that nonetheless produces a poor image when passed to generation, while the generation branch can synthesize a visually appealing image from which the original layout cannot be recovered. Without a verifiable link between directions, joint optimization remains unstable. We present CyCLeGen, a **Cy**cle-**C**onsistent **L**ayout-based autoregressive framework that mutually couples layout understanding and layout-to-image generation through bidirectional cycle-consistency (Figure 2). The core idea is simple: a predicted layout must be able to generate a faithful image, and a generated image must be parseable back into the original layout. This cycle-consistent constraint converts the two competing objectives into a single coherent learning signal.

To enable joint optimization of two inverse tasks, we introduce **CycleGRPO**, a bidirectional reinforcement learning interleave training strategy that alternates between two directions: (1) in the understanding$\rightarrow$generation direction, the model rolls out layout predictions and evaluates them through downstream image generation quality (IoU + CLIP rewards); (2) in the generation$\rightarrow$understanding direction, the model rolls out image generations and evaluates whether

*Equal contribution.

†Work done during internship at UC San Diego.

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

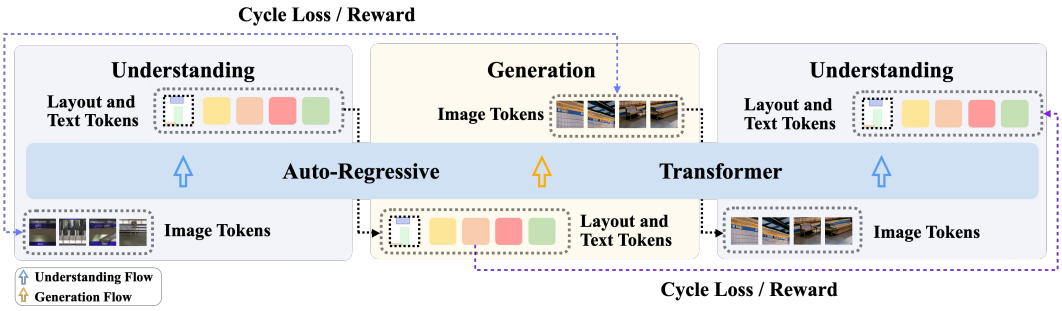

Figure 1: **Overview of the CyCLeGen framework.** CyCLeGen employs a unified autoregressive transformer that jointly performs layout understanding (image → layout) and layout-conditioned image generation (layout → image) within a shared token space. Given image tokens, the model predicts layout and text tokens through the understanding branch; conversely, conditioned on layout and text tokens, the generation branch produces image tokens. Training enforces bidirectional cycle consistency across the two directions, forming image → layout → image and layout → image → layout loops. Cycle rewards/losses encourage the generated outputs in one direction to remain structurally and semantically recoverable in the reverse direction, aligning visual understanding with controllable image generation.

their layouts can be recovered (IoU + HPS rewards). Both directions share the same backbone but update complementary parameters, forming a self-reinforcing cycle.

This design has two notable properties. First, *self-introspection*: because one direction evaluates the output of the other, the model receives automatic feedback on its own generations without external supervision. In this formulation, generation helps understanding by serving as a downstream verifier, a layout is rewarded only if it produces a high-quality image; symmetrically, understanding helps generation by acting as a structural auditor, an image is rewarded only if its layout can be recovered. This mutual introspection mechanism provides concrete, gradient-carrying evidence that the two directions benefit each other, rather than merely sharing parameters. Second, *data efficiency*: the cycle reward operates on the model's own predictions, effectively creating synthetic supervision from only 8k RL samples, far fewer than the millions of paired examples typically required. Crucially, we show that *removing* the cycle constraint from RL causes severe degradation across all metrics, confirming that the mutual recoverability signal, not RL alone, is the key driver of improvement.

Our contributions are threefold:

- We introduce cycle-consistency between layout understanding and layout-to-image generation within a single autoregressive model, enabling self-introspective learning from only 8k RL samples.
- We propose *CycleGRPO*, a bidirectional reinforcement learning strategy that uses complementary geometric and perceptual rewards to jointly optimize both directions without task interference.
- We show consistent improvements in both layout understanding and generation, together with emergent gains on untrained captioning.

2 Related Work

Layout Understanding and Layout-to-Image Generation Layout-to-image generation has been explored through both training-free approaches [3, 4] and training-based diffusion models [5–11]. Layout is also widely used as a proxy for general generation scenario[12–15]. While these methods improve spatial controllability, they operate solely in the *generation* direction and do not learn how layouts map back from images. PlanGen[1] is among the first attempts to unify layout understanding and generation in an autoregressive model, but its formulation is still limited: understanding and generation remain loosely coupled, lacking mutual constraints. Our method addresses these limitations by enforcing *cycle-consistency* between layout to image and image to layout, enabling stronger structural correctness and verifiable feedback between the two tasks.

Reinforcement Learning for Multimodal Models Reinforcement learning has become central to advancing reasoning in large language models, with notable progress in math [16–18], coding [19, 20], and pure-RL systems such as DeepSeek-R1-Zero [21]. In vision–language settings, RL has primarily targeted hallucination mitigation and preference alignment [22–24]; recent unified architectures such

as Show-o [25], Transfusion [26], Janus-Pro [2], Bagel [27], and UniGen [28] demonstrate that understanding and generation can coexist in one transformer, yet none employs RL with *verifiable cross-directional rewards* that explicitly link the two capabilities. Our CycleGRPO fills this gap by providing cycle-aware, automatically verifiable rewards for both layout understanding and layout-conditioned generation, turning the inverse-task structure itself into a supervision signal.

Introspective Learning The framework of energy-based generative adversarial learning was first proposed in [29], with a single unified model being both generative and discriminative. This approach was subsequently evolved into Introspective Neural Networks (INN) [30–32]. Analogous to how INN uses a single model to both generate and critique samples, our cycle-consistency enables the model to evaluate its own generations by re-parsing them, but operates at the level of structured layout–image pairs rather than pixel-level classification.

3 Method

3.1 Overview

CyCLeGen aims to improve the cycle-consistent alignment between the layout prediction and the layout-conditioned generation within a unified model. We adopt a training strategy that explicitly enforces cycle consistency across the framework through supervised fine-tuning (SFT) warm-up and cycle-consistent reinforcement learning (RL). Notably, the layout input used in our pipeline contains both precise bounding-box annotations and fine-grained textual descriptions for each object instance.

3.2 Supervised Fine-tuning Warm-Up

We first perform cycle-consistent supervised fine-tuning as a *warm-up* stage to enable the model to reason over its own generated structures. Rather than pursuing performance gains through extensive supervised training with massive data, the SFT phase merely initializes the model to operate in the cycle input and output format required by subsequent RL.

Specifically, given an input image I and its corresponding layout annotation L, the understanding objective minimizes the token-level cross-entropy between the predicted and ground-truth layout tokens. To close the cycle, instead of conditioning generation on the ground-truth layout[1], we explicitly feed the *predicted* layout $\hat{L}$ from the understanding branch into the generation module. The generation objective minimizes the token-level cross-entropy between the predicted and ground-truth image tokens. The SFT objective is defined as the sum of the understanding loss and the generation loss, i.e., $L_{\mathrm{SFT}} = L_{\mathrm{Und}} + L_{\mathrm{Gen}}$, ensuring both understanding and generation modules share a consistent representation space before RL.

3.3 Cycle-consistent Reinforcement Learning

Different from prior RL approaches that optimize either visual generation or visual understanding in isolation, we introduce **CycleGRPO**, a bidirectional cycle-consistent GRPO framework for interleaved multimodal modeling. Our formulation jointly optimizes the two complementary reasoning cycles: layout→image→layout generation and image→layout→image. By interleaving these two cycle directions during training, the model learns not only to produce high-quality images or accurate layouts, but also to ensure that the generated visual structures remain semantically recoverable. This cycle constraint provides a form of *self-introspection* between generation and understanding, which stabilizes reinforcement learning in unified multimodal models.

GRPO formulation We build on Group Relative Policy Optimization (GRPO) [21, 16], which estimates advantages without a learned critic. Given input x, the policy samples G candidate outputs $\{o_1, \ldots, o_G\}$ and scores them with a reward function. The advantage of each output is computed by normalizing within the group:

$$A_i = \frac{r_i - \mathrm{mean}(\{r_j\}_{j=1}^{G})}{\mathrm{std}(\{r_j\}_{j=1}^{G})}. \tag{1}$$

The policy is updated via a clipped surrogate objective with KL regularization (details in [21]).

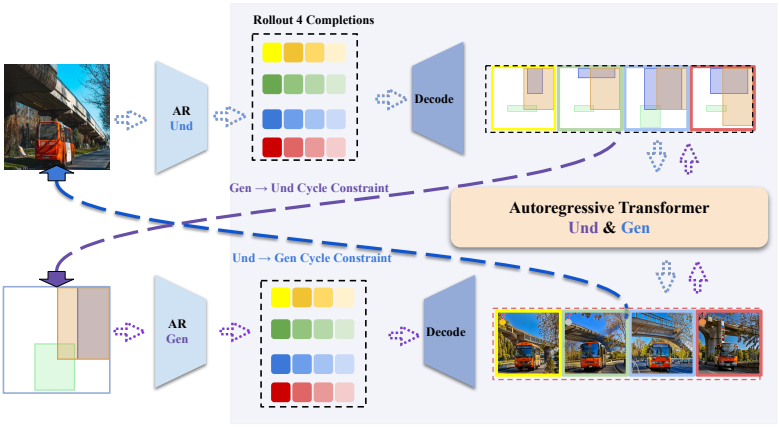

Figure 2: **Overview of CycleGRPO.** CycleGRPO trains a unified autoregressive transformer for both layout *understanding* and layout-conditioned *generation* through two complementary directions. The **Und** $\rightarrow$ **Gen** direction (blue arrows) starts from an input image, where the model autoregressively predicts layouts via trajectory rollouts and decodes them into bounding boxes. The **Gen** $\rightarrow$ **Und** direction (purple arrows) starts from a layout prompt, where the model generates images and enforces that the generated images can recover the original layout through understanding. Multiple trajectories are sampled in both directions and optimized with GRPO, providing cycle-consistent reinforcement signals that align visual understanding with image generation.

Importantly, because GRPO operates solely on sampled outputs and scalar rewards—without requiring task-specific critics—it applies naturally to both directions of our unified model: **layout$\rightarrow$image generation** (optimizing visual quality and structural consistency) and **image$\rightarrow$layout understanding** (optimizing layout accuracy and interpretability). This symmetry enables an **interleaved RL optimization**, where rollout occurs alternately in the understanding and generation pathways under complementary reward signals. In practice, we use separate optimizer groups for different directions, denoted as θ_{Und} and θ_{Gen}, which share the unified AR backbone and differ only in their generation and understanding encoder/decoder, without introducing any model-level modularity.

Understanding $\rightarrow$ Generation. In this direction, the model first performs layout prediction, where multiple layout candidates are generated via rollout. The predicted layouts are then fed into the layout-to-image generation module, which proceeds to layout-to-image generation. The reward function evaluates both the structural accuracy of the predicted layouts and the perceptual quality of the generated images:

$$R_1 = \lambda_{\text{iou}} \cdot \text{IoU}(L_{\text{pred}}, L_{\text{gt}}) + \lambda_{\text{clip}} \cdot \text{CLIP}(I_{\text{pred}}, I_{\text{gt}}), \tag{2}$$

where IoU measures geometric consistency between predicted and ground-truth layouts, and CLIP quantifies semantic consistency between generated and reference images. We optimize the θ_{Und} via:

$$\theta_{\text{Und}} \leftarrow \theta_{\text{Und}} - \eta \nabla_{\theta_{\text{Und}}} \mathcal{L}^{(1)}_{\text{GRPO}}, \tag{3}$$

Generation $\rightarrow$ Understanding. In this direction, the model first performs layout-to-image generation by rolling out multiple image candidates conditioned on the input layout. The generated images are then passed to the understanding module to predict layouts. This reverse cycle encourages the generation module to synthesize images whose structural content can be reliably recovered. The reward therefore emphasizes both high-fidelity image synthesis and semantic recoverability:

$$R_2 = \lambda_{\text{iou}} \cdot \text{IoU}(L_{\text{pred}}, L_{\text{gt}}) + \lambda_{\text{hps}} \cdot \text{HPSv2}(I_{\text{pred}}), \tag{4}$$

where IoU similarly regularizes regulate the layout position and HPSv2 is a high-level perceptual similarity metric capturing human-aligned aesthetic and semantic quality. We optimize the θ_{Gen} via:

$$\theta_{\text{Gen}} \leftarrow \theta_{\text{Gen}} - \eta \nabla_{\theta_{\text{Gen}}} \mathcal{L}^{(2)}_{\text{GRPO}}, \tag{5}$$

Cycle-consistent GRPO Objective. For each direction, the GRPO loss optimizes the model toward higher reward preference via gradient regularized updates. R denotes the reward assigned to the generated output, while b is a baseline value:

$$\mathcal{L}_{\text{GRPO}} = -\mathbb{E}_{(x,y)}\big[\log P_\theta(y \mid x) \cdot (R - b)\big] \tag{6}$$

4 Experiment

4.1 Experiment Setup

Training For fair comparison, we sample our training data on LayoutSAM [9], which provides not only bounding-box annotations but also rich textual descriptions for each object instance. This combination of structured geometry and detailed semantics offers a more informative supervision signal than layouts alone. Following PlanGen [1], all images are resized to 384×384, and CycleGRPO training uses only 8K samples in 1000 steps.

Evaluation We quantitatively evaluate our model on all three key capabilities: *image captioning*, *layout understanding* (Und), and *layout-to-image generation* (Gen). All evaluations are conducted on the held-out test splits of LayoutSAM and OverlayBench, demonstrating consistent improvements across caption quality, structural understanding accuracy, and layout-conditioned generation fidelity.

4.2 Quantitative and Qualitative Results on Image Generation

4.2.1 OverlayBench

Table 1: Comparison of diffusion-based and autoregressive **layout-to-image generation** models on the **OverlayBench** [33] benchmark across Simple, Regular, and Complex splits. Improvements are computed relative to PlanGen.

Type	Method	# Params	mIoU(%) ↑	O-mIoU(%) ↑	SR_E(%) ↑	SR_R(%) ↑	$CLIP_{Global}$ ↑	$CLIP_{Local}$ ↑
			OverlayBench-Simple					
	GLIGEN [5]	1.07B	60.54	36.22	49.99	78.72	34.17	24.75
	InstanceDiff [7]	1.23B	**71.21**	**49.99**	77.71	87.49	34.25	27.69
Diffusion	MIGC [8]	0.86B	58.64	32.15	63.41	81.60	33.07	26.49
	HiCo [6]	1.22B	69.47	47.23	67.75	86.08	35.25	27.04
	CreatiLayout [9]	3.31B	58.78	32.52	72.34	84.45	**37.29**	27.49
	PlanGen [1]	1.5B	62.94	38.54	82.62	90.53	35.62	27.73
AR	SFT Warm-up	1.5B	63.02 (+0.08)	38.70 (+0.16)	82.80 (+0.18)	90.56 (+0.03)	35.43 (-0.19)	27.77 (+0.04)
	RL w/o Cycle	1.5B	57.02 (-5.92)	31.26 (-7.28)	71.38 (-11.24)	89.82 (-0.71)	34.05 (-1.57)	25.65 (-2.08)
	CyCLeGen (ours)	1.5B	64.84 (+1.90)	42.03 (+3.49)	**85.05** (+2.43)	**90.53** (+0.00)	35.48 (-0.14)	**27.89** (+0.16)
			OverlayBench-Regular					
	GLIGEN [5]	1.07B	52.46	26.53	44.88	77.46	33.93	23.42
	InstanceDiff [7]	1.23B	**60.08**	**34.15**	72.51	83.36	33.09	26.19
Diffusion	MIGC [8]	0.86B	47.42	20.06	56.67	77.85	32.72	24.99
	HiCo [6]	1.22B	55.02	29.60	58.24	79.89	33.91	25.34
	CreatiLayout [9]	3.31B	47.04	20.67	62.60	78.31	**36.67**	25.55
	PlanGen [1]	1.5B	53.24	27.57	78.02	87.40	35.03	26.25
AR	SFT Warm-up	1.5B	53.32 (+0.08)	27.72 (+0.15)	78.18 (+0.16)	87.44 (+0.04)	34.60 (-0.43)	26.30 (+0.05)
	RL w/o Cycle	1.5B	46.37 (-6.87)	21.02 (-6.55)	65.76 (-12.26)	86.12 (-1.28)	33.18 (-1.85)	24.46 (-1.79)
	CyCLeGen (ours)	1.5B	54.21 (+0.97)	29.04 (+1.47)	**80.58** (+2.56)	**87.98** (+0.58)	34.74 (-0.29)	**26.42** (+0.17)
			OverlayBench-Complex					
	GLIGEN [5]	1.07B	50.79	23.85	41.70	79.93	33.92	22.75
	InstanceDiff [7]	1.23B	**53.68**	**25.63**	66.02	80.34	32.33	25.53
Diffusion	MIGC [8]	0.86B	40.04	13.26	47.80	74.48	31.93	24.20
	HiCo [6]	1.22B	46.56	20.35	48.88	75.19	33.15	24.41
	CreatiLayout [9]	3.31B	44.24	18.05	52.10	79.98	**36.55**	24.76
	PlanGen [1]	1.5B	49.87	22.94	72.82	86.80	35.09	25.37
AR	SFT Warm-up	1.5B	49.94 (+0.07)	23.05 (+0.11)	72.95 (+0.13)	86.84 (+0.04)	34.95 (-0.14)	25.44 (+0.07)
	RL w/o Cycle	1.5B	43.69 (-6.18)	17.08 (-5.86)	56.71 (-16.11)	87.13 (+0.33)	33.07 (-2.02)	23.53 (-1.84)
	CyCLeGen (ours)	1.5B	50.72 (+0.85)	24.50 (+1.56)	**74.01** (+1.19)	**87.07** (+0.27)	34.57 (-0.52)	**25.58** (+0.21)

To comprehensively assess the robustness of layout-to-image (L2I) generation models under challenging real-world conditions, we conduct detailed evaluations on the OverlayBench [33] benchmark, which provides three difficulty splits—Simple, Regular, and Complex—derived from the OverLayScore that quantifies spatial clutter, object density, and semantic ambiguity. While prior benchmarks such as COCO[34] and LayoutSAM[9] largely feature well-separated objects and low-overlap

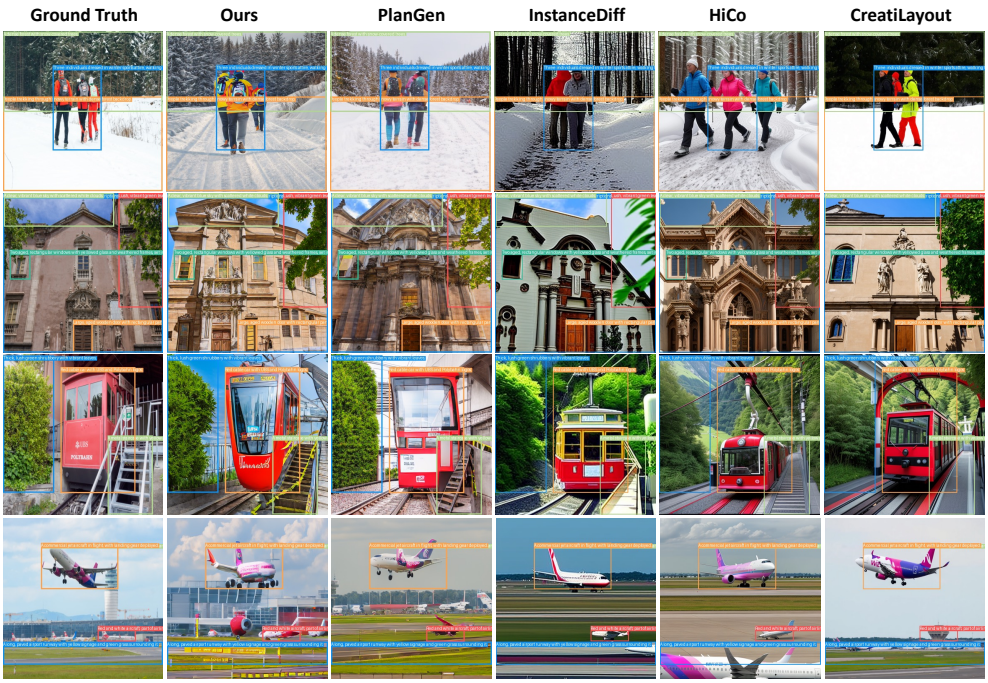

Figure 3: **Qualitative comparison of layout-to-image generation.** Given the same input layouts, our CyCLeGen model produces images that more faithfully preserve object geometry, spatial configuration, and fine-grained structure compared to PlanGen and diffusion-based baselines (InstanceDiff[7], HiCo[6], CreatiLayout[9]). Across diverse scenes—including people, architectural facades, cable cars, and airplanes—CyCLeGen demonstrates stronger adherence to layout constraints and generates visually coherent, detail-rich images.

layouts, their limited structural diversity makes it difficult to stress-test the spatial reasoning and fine-grained grounding capabilities of modern L2I systems. In contrast, OverlayBench is specifically curated to expose systematic weaknesses: its harder splits contain dense multi-object compositions with substantial bounding box overlap, frequent inter-object occlusion, and high semantic similarity between co-occurring instances.

Overlaybench introduces several challenging indicators. For instance, O-mIoU (Overlap mIoU) computes the mIoU restricted to ground-truth overlap regions and their corresponding predicted regions. By isolating areas shared between instances, O-mIoU provides a more sensitive and discriminative measure of fidelity in occluded or entangled scenarios than standard global mIoU. SRR (Success Rate of Relationship) measures the percentage of object pairs whose predicted spatial relationships match the ground truth, offering an interpretable, relationship-level metric of success. SRE (Success Rate of Entity) measures the percentage of entities whose predicted attributes (e.g., category, location, or existence) correctly match the ground truth. These characteristics create severe challenges for both diffusion-based methods (e.g., GLIGEN [5], InstanceDiff [7], MIGC [8], HiCo [6], CreatiLayout [9]) and autoregressive approaches such as PlanGen [1].

As shown in table 1, our CycleGRPO reinforcement learning strategy enhances structure preservation and relation fidelity, particularly on the Regular and Complex splits where spatial entanglement is most pronounced. These results demonstrate that the proposed cycle-consistent training pipeline not only mitigates common failure modes under severe layout complexity, but also closes the performance gap between autoregressive and strong diffusion-based models in highly entangled scenes. For PlanGen, we use the official pretrained 1.5B checkpoint and the same layout serialization, image resolution, decoding temperature, and sampling strategy as CyCLeGen. Since PlanGen produces 384×384 images, we resize all AR outputs to 512×512 before computing OverlayBench metrics, matching the resolution expected by the benchmark and diffusion baselines.

We further investigate the effect of removing the cycle-consistency constraint during reinforcement learning. Without the bidirectional cycle supervision between layout understanding and layout-conditioned image generation, the RL optimization tends to focus on improving individual objectives while neglecting the structural alignment between the two tasks. As a result, the model exhibits no-

ticeable performance degradation across multiple metrics compared with the PlanGen baseline. This observation highlights that naive reinforcement learning without the cycle constraint can introduce optimization instability and task interference, ultimately harming the model's overall performance. In contrast, our cycle-consistent training explicitly enforces mutual consistency between understanding and generation, providing a stabilizing signal that leads to consistent improvements. Note that SFT alone brings marginal improvement, consistent with its role as a format warm-up rather than a performance-driven training stage; CycleGRPO is responsible for the dominant gains.

Table 2: **Layout-to-image** quantitative comparison on **LayoutSAM-Eval [9]**. We report region-wise quality (Spatial, Color, Texture, Shape) and global perceptual metrics (PickScore, FID, IS, HPSv2).

Type	Method	Region-wise Quality				Global-wise Quality			
		Spatial ↑	Color ↑	Texture ↑	Shape ↑	Pick ↑	FID ↓	IS ↑	HPSv2 ↑
	Real Images	98.95	98.45	98.90	98.80	-	-	-	-
Diffusion	GLIGEN [5]	77.53	49.41	55.29	52.72	20.78	21.92	20.57	24.15
	InstanceDiff [7]	87.99	69.16	72.78	71.08	21.01	19.67	20.02	21.25
	MIGC [8]	85.66	66.97	71.24	69.06	20.71	21.19	19.65	23.43
	HiCo [6]	90.22	69.89	73.95	72.75	**22.44**	19.30	**22.50**	**27.51**
	CreatiLayout [9]	92.67	74.45	77.21	75.93	22.02	19.10	22.04	25.95
AR	PlanGen [1]	92.21	82.69	86.53	85.36	21.43	**13.91**	19.31	24.50
	CyCLeGen (ours)	**92.71**	**84.22**	**87.97**	**87.02**	21.53	15.62	20.42	25.96

4.2.2 LayoutSAM-eval

To evaluate fine-grained controllability and structural fidelity in layout-to-image generation, we benchmark diffusion-based and autoregressive models on LayoutSAM-Eval [9]. Beyond standard global perceptual metrics such as CLIP [35], FID [36], IS[37], and HPSv2 [38], this benchmark measures *region-wise* consistency across Spatial, Color, Texture, and Shape attributes, providing a more sensitive diagnostic of how faithfully a model adheres to object-level layout constraints.

Diffusion models demonstrate strong overall image quality but still exhibit noticeable inconsistencies in regional appearance, especially under dense layouts with fine object boundaries. In contrast, autoregressive models inherently maintain token-level structural coupling, enabling more precise alignment between layout tokens and generated visual attributes. Our method further strengthens this alignment: despite using a unified AR framework, it achieves the highest scores across all four regional metrics, surpassing both diffusion-based baselines and the autoregressive PlanGen[1]. These results highlight the effectiveness of our cycle-consistent training and RL-enhanced layout grounding in preserving detailed object-level semantics. We further present qualitative layout-to-image generation results in fig. 3, which compares our method against PlanGen, InstanceDiff, HiCo, and CreatiLayout. Across diverse scenes, CyCLeGen produces images that more faithfully adhere to the input layouts, preserving object geometry, spatial relations, and appearance details. This result highlights the effectiveness of our framework. We note a modest increase in FID, since cycle-consistency prioritizes structural fidelity and region-wise consistency over pixel-level distributional matching. The substantial gains on all four region-wise metrics confirm that our model better preserves object-level layout constraint, which is the primary goal of layout-conditioned generation.

4.3 Quantitative and Qualitative Results on Image Understanding

4.3.1 Layout Understanding

Following Plangen[1], to assess the effectiveness of our approach in layout understanding, we evaluate the model on the LayoutSamEval benchmark. In addition to the original PlanGen, we incorporate Grounding DINO and Janus Pro as strong detection-oriented baselines. Grounding DINO offers a high-quality object localization reference point, while Janus Pro represents a competitive multimodal large language model with robust grounding capabilities. As shown in table 3, CyCLeGen yields higher AP and dramatically improved AR, demonstrating stronger grounding completeness and more reliable extraction of layout structures. While AP emphasizes how accurate and well-ranked the predictions are, AR emphasizes whether the model finds all relevant objects at all. AR

Table 3: Quantitative results on **layout understanding** evaluated on the LayoutSamEval benchmark[9]. We compare our model with the PlanGen baseline, along with strong vision-language and detection baselines, including Grounding-DINO [39] and Janus-Pro[2]. Metrics include AP, AP50, AP75, and AR. Improvements are computed relative to PlanGen.

Setting	AP↑	AP50↑	AP75↑	AR↑
GroundingDINO [39]	42.53	47.20	44.60	83.61
Janus-Pro [2]	9.05	13.68	7.20	14.67
PlanGen-baseline [1]	27.66	33.94	30.56	51.58
CyCLeGen (ours)	28.33 (+0.67)	36.99 (+3.05)	30.81 (+0.25)	71.05 (+19.47)

is therefore particularly informative in proposal-based, layout, and grounding-oriented detection settings, where complete spatial coverage is more critical than confidence ordering. These results confirm that our progressive training strategy not only enhances layout understanding within the autoregressive modeling framework but also achieves stronger overall performance. The moderate AP improvement is partly attributable to annotation noise in LayoutSAM, whose labels are auto-generated by Grounding-DINO rather than manually verified or state-of-the-art vision language models. We adopt this dataset for fair comparison with PlanGen. As illustrated in fig. 4, our model generates significantly more accurate and complete layout predictions than the PlanGen baseline. This indicates that our proposed cycle-consistent training effectively reduces missed detections, stabilizes bounding-box localization, and produces semantically coherent descriptions even in crowded or highly structured scenes.



Figure 4: **Qualitative comparison of layout understanding results.** We compare CyCLeGen with PlanGen and GroundingDINO across diverse scenes. CyCLeGen produces more complete and precise detections, with fewer missed objects, cleaner bounding boxes, and more semantically coherent structured descriptions. In contrast, PlanGen often yields fragmented boxes, duplicated predictions, and under-detection in complex scenes. These results demonstrate that our cycle-consistent training substantially boosts structural grounding quality.

4.3.2 Image Captioning as An Emergent Task

Table 4: Quantitative comparison of the **image captioning** performance on the LayoutSamEval benchmark[9]. We report BLEU-1/2/3/4, METEOR, ROUGE-L, and CIDEr scores for the baseline PlanGen [1] and Janus Pro [2], our SFT Warm-up, and CyCLeGen. Note that CyCLeGen has **not been trained on** this captioning task, yet, it demonstrates a substantial improvement over the baselines, clearly demonstrating the strong advantage of adding the cycle-consistency to the basic visual unified model. Improvements are computed relative to PlanGen.

Method	BLEU-1↑	BLEU-2↑	BLEU-3↑	BLEU-4↑	METEOR↑	ROUGE-L↑	CIDEr↑
Janus-Pro [2]	35.26	19.73	11.19	6.66	21.78	26.04	12.80
PlanGen-baseline [1]	43.06	30.20	22.68	17.92	21.92	33.81	37.48
SFT Warm-up	51.72 (+8.66)	37.97 (+7.77)	29.27 (+6.60)	23.58 (+5.66)	25.43 (+3.51)	40.10 (+6.29)	57.90 (+20.42)
RL w/o cycle	46.63 (+3.57)	31.83 (+1.63)	23.20 (+0.52)	17.99 (+0.07)	21.85 (-0.07)	34.92 (+1.11)	38.86 (+1.38)
CyCLeGen (ours) RL	52.99 (+9.93)	39.91 (+9.71)	31.41 (+8.73)	25.68 (+7.76)	26.84 (+4.92)	42.35 (+8.54)	66.46 (+28.98)

To demonstrate the **cross-task generalization** capability of CyCLeGen, we evaluate it on image captioning, an emergent task that is **not included** in the reinforcement learning optimization. As shown in table 4, on the LayoutSamEval benchmark, the model achieves consistent improvements as it progresses from the original PlanGen baseline to the supervised fine-tuning (SFT) variant and then to the two-stage reinforcement learning procedure. This trend highlights the effectiveness of CyCLeGen in transferring improvements beyond the optimized tasks, and the ablation study further confirms the benefit of introducing the cycle-consistency loss.

Specifically, SFT substantially improves lexical and structural alignment, as reflected by gains in BLEU-1/2/3/4 and ROUGE-L. Building on this, RL further enhances semantic fidelity, yielding stronger METEOR and CIDEr scores, which indicate more descriptive and human-aligned captions. CyCLeGen also achieves the best overall performance across all metrics, demonstrating that combining cycle-consistent layout understanding with GRPO-based policy optimization leads to stronger captioning ability. These results show that jointly improving image understanding and image generation can bring **general and universal benefits to each other**.

Table 5: Comparison with PlanGen and our variants on **LayoutSAM-Eval[9]**. Understanding metrics are on the left, and generation metrics on the right.

Model	AP↑	AP50↑	AP75↑	AR↑	Spatial↑	Color↑	Texture↑	Shape↑
PlanGen	27.66	33.94	30.56	51.58	92.21	82.69	86.53	85.36
PlanGen-Und	28.00	37.78	29.40	48.17	88.16 (-4.05)	76.92 (-5.77)	80.71 (-5.82)	79.71 (-5.65)
PlanGen-Gen	8.41 (-19.25)	16.33 (-17.61)	7.27 (-23.29)	17.62 (-33.96)	91.41	81.83	85.84	84.81
PlanGen-Joint	24.28 (-3.38)	34.26 (+0.32)	25.38 (-5.18)	45.54 (-6.04)	91.00 (-1.21)	81.75 (-0.94)	85.64 (-0.89)	84.28 (-1.08)
RL-w/o cycle	27.83 (+0.17)	36.77 (+2.83)	29.63 (-0.93)	51.61 (+0.03)	91.72 (-0.49)	82.45 (-0.24)	86.78 (+0.25)	86.19 (+0.83)
CyCLeGen (ours)	**28.33** (+0.67)	**36.99** (+3.05)	**30.81** (+0.25)	**71.05** (+19.47)	**92.71** (+0.50)	**84.22** (+1.53)	**87.97** (+1.44)	**87.02** (+1.66)

4.4 Ablation

Our ablation results further demonstrate the necessity of cycle-consistent modeling. table 5 summarizes different training configurations compared with the PlanGen baselines: PlanGen-Und fine-tunes the model exclusively on the visual understanding objective, while PlanGen-Gen fine-tunes on the layout-to-image generation objective in a next token prediction paradigm. PlanGen-Joint applies interleaved joint training of understanding and generation just as Plangen baseline. PlanGen-w/o cycle adopts the same training strategy as us, which builds on SFT warm-up and RL training (IOU reward for Layout Understanding and HPSv2 for layout-to-image Generation)for understanding and generation without explicitly enforcing cycle consistency. The table first reveals a clear phenomenon of SFT forgetting: optimizing the model for visual understanding improves detection but degrades generation fidelity (PlanGen-Und), while optimizing for generation severely weakens visual understanding (PlanGen-Gen). This trade-off highlights the limitation of naive or one-sided fine-tuning strategies. Notably, although PlanGen-w/o cycle partially alleviates this issue by jointly applying RL on top of an SFT-initialized model, it still fails to consistently preserve bidirectional competence and even encounters a forgetting problem over baselines, indicating that applying reinforcement learning optimization alone is insufficient. In contrast, explicitly enforcing cycle consistency on the model's own predictions (CyCLeGen), rather than relying on simply interleaved joint supervised finetuning (PlanGen-Joint) or solely reinforcement learning training(RL-w/o cycle), leads to consistent and robust improvements over an already **converged** PlanGen baselines. Although RL training incurs higher per-step overhead, the data efficiency afforded by cycle-consistency (only 8k samples) keeps the total CycleGRPO training cost below half the GPU hours required by joint supervised fine-tuning.

5 Conclusion

We presented CyCLeGen, a unified framework that introduces cycle-consistency to unified models for image understanding and generation within integrated autoregressive formulation. By introducing CycleGRPO reinforcement strategy, our model reduces the mutual interference that commonly appears in unified vision–language systems and achieves more stable bidirectional alignment. Experiments across layout prediction, controllable generation, and emergent captioning tasks show clear and consistent improvements over the PlanGen baseline, highlighting the effectiveness and generalization capabilities of enforcing reciprocal supervision between inverse tasks. We believe our cycle-consistent formulation offers a simple yet powerful direction for building unified multimodal models that can better align, self-correct, and generalize across understanding and generation tasks.

Acknowledgment

Acknowledgement. This work is supported by U.S. National Science Foundation Award IIS-2433768 and IIS-2127544. This work also used DeltaAI at National Center for Supercomputing Applications (NCSA) through allocation CIS260180 from the Advanced Cyberinfrastructure Coordination Ecosystem: Services & Support (ACCESS) program, which is supported by U.S. National Science Foundation grants #2138259, #2138286, #2138307, #2137603, and #2138296.

References

[1] Runze He, Bo Cheng, Yuhang Ma, Qingxiang Jia, Shanyuan Liu, Ao Ma, Xiaoyu Wu, Liebucha Wu, Dawei Leng, and Yuhui Yin. Plangen: Towards unified layout planning and image generation in auto-regressive vision language models. In *ICCV*, 2025.

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

[3] Minghao Chen, Iro Laina, and Andrea Vedaldi. Training-free layout control with cross-attention guidance. In *WACV*, 2024.

[4] Jinheng Xie, Yuexiang Li, Yawen Huang, Haozhe Liu, Wentian Zhang, Yefeng Zheng, and Mike Zheng Shou. Boxdiff: Text-to-image synthesis with training-free box-constrained diffusion. In *ICCV*, 2023.

[5] Yuheng Li, Haotian Liu, Qingyang Wu, Fangzhou Mu, Jianwei Yang, Jianfeng Gao, Chunyuan Li, and Yong Jae Lee. Gligen: Open-set grounded text-to-image generation. In *CVPR*, 2023.

[6] Bo Cheng, Yuhang Ma, Liebucha Wu, Shanyuan Liu, Ao Ma, Xiaoyu Wu, Dawei Leng, and Yuhui Yin. Hico: Hierarchical controllable diffusion model for layout-to-image generation. *NeurIPS*, 2024.

[7] Xudong Wang, Trevor Darrell, Sai Saketh Rambhatla, Rohit Girdhar, and Ishan Misra. Instancediffusion: Instance-level control for image generation. In *CVPR*, 2024.

[8] Dewei Zhou, You Li, Fan Ma, Xiaoting Zhang, and Yi Yang. Migc: Multi-instance generation controller for text-to-image synthesis. In *CVPR*, 2024.

[9] Hui Zhang, Dexiang Hong, Yitong Wang, Jie Shao, Xinglong Wu, Zuxuan Wu, and Yu-Gang Jiang. Creatilayout: Siamese multimodal diffusion transformer for creative layout-to-image generation. In *ICCV*, 2025.

[10] Xin Lin, Haodong Li, Zhifei Zhang, Yutong Yang, Haitian Zheng, Juanxi Tian, Zhe Lin, and Truong Nguyen. Pixelcontrol: Fine-grained condition fidelity in text-to-image diffusion, 2026. URL `https://arxiv.org/abs/2608.15705`.

[11] Divyansh Srivastava, Xiang Zhang, He Wen, Chenru Wen, and Zhuowen Tu. Lay-your-scene: Natural scene layout generation with diffusion transformers. In *ICCV*, October 2025.

[12] Guanning Zeng, Xiang Zhang, Zirui Wang, Haiyang Xu, Zeyuan Chen, Bingnan Li, and Zhuowen Tu. Yolo-count: Differentiable object counting for text-to-image generation. In *ICCV*, 2025.

[13] Yilin Wang, Haiyang Xu, Xiang Zhang, Zeyuan Chen, Zhizhou Sha, Zirui Wang, and Zhuowen Tu. Omnicontrolnet: Dual-stage integration for conditional image generation. In *CVPRW*, 2024.

[14] Yucheng Mao, Zeyuan Chen, Xiaojun Shan, Xiang Zhang, Divyansh Srivastava, Bingnan Li, and Zhuowen Tu. Quadtok: Quadtree visual tokenizer for autoregressive image generation, 2026. URL `https://arxiv.org/abs/2610.10497`.

[15] Zeyuan Chen, Xiang Zhang, Haiyang Xu, Jianwen Xie, and Zhuowen Tu. Cvp: Central-peripheral vision-inspired multimodal model for spatial reasoning. In *WACV*, 2025.

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

[17] An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, et al. Qwen2. 5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*, 2024.

[18] Huaiyuan Ying, Shuo Zhang, Linyang Li, Zhejian Zhou, Yunfan Shao, Zhaoye Fei, Yichuan Ma, Jiawei Hong, Kuikun Liu, Ziyi Wang, et al. Internlm-math: Open math large language models toward verifiable reasoning. *arXiv preprint arXiv:2402.06332*, 2024.

[19] Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Kai Dang, et al. Qwen2. 5-coder technical report. *arXiv preprint arXiv:2409.12186*, 2024.

[20] Fangkai Jiao, Geyang Guo, Xingxing Zhang, Nancy F. Chen, Shafiq Joty, and Furu Wei. Preference optimization for reasoning with pseudo feedback. In *ICLR*, 2025.

[21] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 2025.

[22] Zhiyuan Zhao, Bin Wang, Linke Ouyang, Xiaoyi Dong, Jiaqi Wang, and Conghui He. Beyond multimodal hallucinations: Enhancing lvlms through hallucination-aware direct preference optimization. In *ICME*, 2025.

[23] Zhiqing Sun, Sheng Shen, Shengcao Cao, Haotian Liu, Chunyuan Li, Yikang Shen, Chuang Gan, Liang-Yan Gui, Yu-Xiong Wang, Yiming Yang, et al. Aligning large multimodal models with factually augmented rlhf. In *ACL*, 2024.

[24] Tianyu Yu, Yuan Yao, Haoye Zhang, Taiwen He, Yifeng Han, Ganqu Cui, Jinyi Hu, Zhiyuan Liu, Hai-Tao Zheng, Maosong Sun, et al. Rlhf-v: Towards trustworthy mllms via behavior alignment from fine-grained correctional human feedback. In *CVPR*, 2024.

[25] Jinheng Xie, Weijia Mao, Zechen Bai, David Junhao Zhang, Weihao Wang, Kevin Qinghong Lin, Yuchao Gu, Zhijie Chen, Zhenheng Yang, and Mike Zheng Shou. Show-o: One single transformer to unify multimodal understanding and generation. In *ICLR*, 2025.

[26] Chunting Zhou, Lili Yu, Arun Babu, Kushal Tirumala, Michihiro Yasunaga, Leonid Shamis, Jacob Kahn, Xuezhe Ma, Luke Zettlemoyer, and Omer Levy. Transfusion: Predict the next token and diffuse images with one multi-modal model. In *ICLR*, 2025.

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

[28] Rui Tian, Mingfei Gao, Mingze Xu, Jiaming Hu, Jiasen Lu, Zuxuan Wu, Yinfei Yang, and Afshin Dehghan. Unigen: Enhanced training & test-time strategies for unified multimodal understanding and generation. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

[29] Zhuowen Tu. Learning generative models via discriminative approaches. In *CVPR*, 2007.

[30] Long Jin, Justin Lazarow, and Zhuowen Tu. Introspective classification with convolutional nets. *NeurIPS*, 2017.

[31] Justin Lazarow, Long Jin, and Zhuowen Tu. Introspective neural networks for generative modeling. In *ICCV*, 2017.

[32] Kwonjoon Lee, Weijian Xu, Fan Fan, and Zhuowen Tu. Wasserstein introspective neural networks. In *CVPR*, 2018.

[33] Bingnan Li, Chen-Yu Wang, Haiyang Xu, Xiang Zhang, Ethan Armand, Divyansh Srivastava, Xiaojun Shan, Zeyuan Chen, Jianwen Xie, and Zhuowen Tu. Overlaybench: A benchmark for layout-to-image generation with dense overlaps. In *NeurIPS*, 2025.

[34] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.

[35] 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 *ICML*, 2021.

[36] 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. In *NeurIPS*, 2017.

[37] 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.

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

[39] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. In *ECCV*, 2024.

[40] Sadeep Jayasumana, Srikumar Ramalingam, Andreas Veit, Daniel Glasner, Ayan Chakrabarti, and Sanjiv Kumar. Rethinking fid: Towards a better evaluation metric for image generation. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9307–9315. IEEE, 2024.

A Additional Result

A.1 Implementation details.

All experiments are conducted on 8×A6000 GPUs. The SFT phase runs for 7,200 steps with as learning rate of 2×10^{-5}. For GRPO, we set the rollout size to 4, with a per-GPU batch size of 1, and the training time is nearly 40 GPU hours. Reward weights are set to $\lambda_{\text{iou}}=1$,$\lambda_{\text{clip}}=1$, $\lambda_{\text{hps}}=2$ and held constant across all experiments without further tuning.

Our model is built on Janus-Pro [2], a unified vision-language model that supports both multimodal understanding and image generation within a single autoregressive transformer. Its key architectural choice is *decoupled visual encoders*: a separate encoder processes images for understanding tasks (e.g., captioning, VQA), while another encoder handles the generation pathway. This decoupling mitigates the representational conflict between perceiving and synthesizing visual content, yet the two directions remain independently optimized without explicit cross-directional feedback.

A.2 Ablation on SFT

Table 6: Comparison of different training strategies on the **LayoutSAM-Eval[9]**. Understanding metrics are shown in the upper panel, and generation metrics in the lower panel. The best result for each metric is highlighted in bold.

(a) Understanding

Method	AP↑	AP_{50}↑	AP_{75}↑	AR↑	BLEU-4↑	METEOR↑	ROUGE-L↑	CIDEr↑
PlanGen	27.66	33.94	30.56	51.58	17.92	21.92	33.81	37.48
SFT warm-up	27.13	35.67	29.25	56.92	23.58	25.43	40.10	57.90
w/o SFT	27.80	36.33	29.20	64.07	24.48	26.79	41.26	62.52
CyCLeGen	**28.33**	**36.99**	**30.81**	**71.05**	**25.68**	**26.84**	**42.35**	**66.46**

(b) Generation

Method	Spatial↑	Color↑	Texture↑	Shape↑	FID↓	IS↑	HPSv2↑	PickScore↑
PlanGen	92.21	82.69	86.53	85.36	**13.91**	19.31	24.50	21.43
SFT warm-up	91.88	82.41	86.76	85.65	14.33	19.30	24.61	21.36
w/o SFT	**93.19**	84.04	87.69	**87.16**	15.41	20.24	25.58	21.47
CyCLeGen	92.71	**84.22**	**87.97**	87.02	15.62	**20.42**	**25.96**	**21.53**

Table 7: Comparison of different training strategies on the **OverlayBench** [33]. Understanding metrics are shown in the upper panel, and generation metrics in the lower panel. All results are averaged over the Simple, Regular, and Complex splits. The best result for each metric is highlighted in bold.

(a) Understanding

Method	AP↑	AP_{50}↑	AP_{75}↑	AR↑
PlanGen	22.08	28.18	23.79	52.48
SFT warm-up	23.25	29.20	24.87	53.04
w/o SFT	25.71	34.50	27.56	64.83
CyCLeGen	**28.07**	**39.90**	**30.81**	**68.80**

(b) Generation

Method	mIoU(%)↑	O-mIoU(%)↑	SR_E(%)↑	SR_R(%)↑	$CLIP_{Global}$↑	$CLIP_{Local}$↑
PlanGen	55.35	29.68	77.82	88.24	**35.25**	26.45
SFT warm-up	55.43	29.82	77.98	88.28	34.99	26.50
w/o SFT	55.58	31.53	78.67	**88.62**	34.65	**26.72**
CyCLeGen	**56.59**	**31.86**	**79.88**	88.53	34.93	26.63

We investigate the role of the SFT warm-up by comparing PlanGen, SFT-only training, CycleGRPO without SFT, and the full CyCLeGen pipeline. Both understanding and generation performance are evaluated on LayoutSAM-Eval and OverLayBench.

As shown in tables 6 and 7, removing the SFT warm-up and directly applying CycleGRPO to PlanGen (w/o SFT) already leads to improvements in both understanding and generation. In contrast, SFT alone mainly benefits language-related understanding metrics, with more limited improvements in layout alignment. Combining SFT with CycleGRPO yields stronger overall performance, particularly for region-level understanding, while maintaining competitive generation quality. These results suggest that SFT provides an effective initialization for subsequent cycle-consistent RL optimization, whereas CycleGRPO is the primary driver of the performance gains.

A.3 Ablation on reward components.

We further study the contribution of each reward component by removing selected reward terms and varying their relative weights. We also compare against RL without cycle-consistency rewards to examine the importance of the proposed bidirectional optimization.

Table 8: Ablation of reward components and their relative weights on the **OverlayBench** [33]. Understanding metrics are shown in the upper panel, and generation metrics in the lower panel. All results are averaged over the Simple, Regular, and Complex splits. The best result for each metric is highlighted in bold.

(a) Understanding

Configuration	AP↑	AP_{50}↑	AP_{75}↑	AR↑
CyCLeGen	28.07	**39.90**	30.81	68.80
w/o HPS	**28.71**	37.68	**31.24**	**72.12**
w/o IoU-UND	26.64	33.64	27.70	64.02
w/o HPS & IoU-UND	25.34	32.29	27.56	63.98
RL w/o Cycle	24.53	30.16	26.75	59.93
Cycle RW 0.5	27.22	39.39	30.59	63.18
Cycle RW 2	27.65	39.23	29.91	67.71

(b) Generation

Configuration	mIoU(%)↑	O-mIoU(%)↑	SR_E(%)↑	SR_R(%)↑	$CLIP_{Global}$↑	$CLIP_{Local}$↑
CyCLeGen	56.59	31.86	79.88	88.53	34.93	26.63
w/o HPS	56.11	31.94	77.40	88.36	34.38	26.43
w/o IoU-UND	**57.18**	**32.06**	80.56	88.70	34.89	26.73
w/o HPS & IoU-UND	56.42	31.79	78.07	88.40	34.34	26.45
RL w/o Cycle	49.03	23.12	64.62	87.69	33.43	24.55
Cycle RW 0.5	54.54	28.31	77.40	88.29	35.66	25.04
Cycle RW 2	56.35	31.31	**80.65**	**89.06**	**36.03**	**27.53**

As shown in tables 8 and 9, removing the cycle-consistency rewards substantially degrades both understanding and generation performance, suggesting that the proposed cycle objective is essential to the effectiveness of our RL framework. Removing IoU-UND primarily weakens grounding performance, while removing HPS reduces preference-based image quality. Meanwhile, some generation metrics improve when IoU-UND is removed, indicating that the two optimization directions involve complementary trade-offs. Varying the reward weights leads to different strengths across individual metrics, but the overall performance remains competitive. These results highlight the importance of jointly optimizing geometric consistency and perceptual quality through the bidirectional cycle.

Image-quality trade-off. As shown in tables 6 and 9, CyCLeGen improves layout alignment and preference-based image quality, although this is accompanied by a higher FID than PlanGen. Removing HPS improves FID but weakens perceptual-preference performance, reflecting a trade-off between distribution fidelity and human-preference alignment. Introducing an additional CMMD[40] reward further reduces FID while retaining competitive perceptual quality. These results suggest that

Table 9: Ablation of reward components on the **LayoutSAM-Eval[9]**. The upper panel compares different reward configurations, while the lower panel evaluates the effect of CMMD on image quality. The best result for each metric within each panel is highlighted in bold.

(a) Reward-component ablations

Configuration	Spatial↑	Color↑	Texture↑	Shape↑	FID↓	IS↑	HPSv2↑	PickScore↑
CyCLeGen	92.71	84.22	87.97	87.02	15.62	**20.42**	25.96	21.53
w/o HPS	92.48	83.89	87.92	86.72	14.70	20.14	24.88	21.41
w/o IoU-UND	**93.01**	84.49	**88.45**	87.14	15.82	20.05	25.95	21.53
w/o HPS & IoU-UND	92.57	84.10	87.93	86.76	**14.59**	20.03	25.02	21.43
RL w/o Cycle	91.72	82.45	86.78	86.19	22.58	19.88	25.78	21.47
Cycle RW 0.5	92.64	84.48	88.35	**87.18**	15.85	20.21	**26.83**	**21.63**
Cycle RW 2	92.55	**84.58**	88.39	87.06	15.08	20.17	25.30	21.56

(b) Controlling distribution fidelity with CMMD[40]

Configuration	FID↓	IS↑	HPSv2↑	PickScore↑
PlanGen	13.91	19.31	24.50	21.43
CyCLeGen	15.62	**20.42**	**25.96**	**21.53**
w/o HPS	14.70	20.14	24.88	21.41
+CMMD	**13.12**	19.65	25.47	21.21

the reward formulation can be adapted to emphasize different aspects of generation quality without changing the overall framework.

Generalization across annotation pipelines. We further evaluate CyCLeGen on OverLayBench to examine whether the understanding improvements depend on the annotation pipeline used for training. Unlike LayoutSAM-Eval, OverLayBench contains compositional overlapping layouts with a different annotation process. As shown in table 7, CyCLeGen consistently outperforms PlanGen and the SFT-only baseline on this unseen benchmark. This suggests that the improvements from cycle-consistent RL are not restricted to the training annotation setting and can transfer to different layout distributions.

A.4 Quantitative Result for Layout Planning and Image Generation

Method	Pick↑	FID ↓	IS↑
GT Layout	21.53	15.62	20.42
PlanGen	21.40	16.18	20.18
CyCLeGen	21.32	19.27	**21.05**

Table 10: Layout planning quantitative comparison of CyCLeGen with Plangen[1] and GT layout on LayoutSAM-Eval[9].

Beyond layout understanding and layout-to-image generation, we further evaluate whether our model can generalize to the layout planning task, despite not receiving any task-specific training or supervision for planning. As shown in table 10, our method achieves a competitive PickScore and clearly improves the Inception Score (IS) over PlanGen. Notably, PlanGen is explicitly trained with layout planning data, whereas our model learns no planning objective at all—yet it still produces high-quality layouts that align well with the ground-truth distribution. This highlights the strong inductive bias introduced by our cycle-consistent training: once the model acquires robust bidirectional layout–image reasoning, it naturally inherits the ability to plan plausible layouts without dedicated supervision.

Table 11: Comparison of image editing performance between PlanGen and CyCLeGen. Higher is better for all metrics.

Method	LocalCLIP ↑	IS ↑	SR(%) ↑
PlanGen	25.28	9.08	14.50
CyCLeGen	**25.31**	**9.18**	**17.00**

Table 12: Object removal results on 200 samples from the COCO test set. Higher is better for all metrics, except for LocalCLIP.

Method	LocalCLIP ↓	IS ↑	SR (%) ↑
PlanGen	23.08	**9.37**	**87.50**
CyCLeGen	**22.92**	9.09	86.00

A.5 Image Editing and Object Removal Evaluation

We evaluate the image editing capability of our cycle-based framework by comparing **PlanGen** and CyCLeGen on two representative tasks: *general image editing* and *object removal*. All methods are evaluated under identical generation settings to ensure a fair comparison.

Following the evaluation protocol adopted in PlanGen, we report three complementary metrics: (1) **LocalCLIP**, which measures the semantic alignment between edited local regions and the corresponding textual instructions, reflecting locality-aware controllability; (2) **Inception Score (IS)**, which evaluates the overall visual quality and realism of the generated images; (3) **Success Rate (SR)**, which measures whether the intended editing objective is successfully achieved. In particular, SR is computed using MiniCPM as an *LLM-as-a-Judge*, where the model is prompted to determine whether the desired editing effect (e.g., object removal or attribute modification) is correctly realized in the generated image.

Image Editing Results. As shown in table 11, CyCLeGen consistently outperforms PlanGen across all metrics on the general image editing task. Notably, CyCLeGen achieves a higher SR, indicating a more reliable execution of editing instructions. Meanwhile, improvements in LocalCLIP and IS demonstrate that the cycle-consistent optimization not only enhances semantic alignment within edited regions but also preserves or slightly improves overall image quality. These results suggest that enforcing a closed-loop consistency between layout planning and image generation effectively mitigates instruction drift and partial edit failures, which are common in open-ended image editing scenarios.

Object Removal Results. We further evaluate object deletion performance on 200 object removal cases derived from the COCO test set. As reported in table 12, both PlanGen and CyCLeGen achieve strong performance, with high SR values, indicating that both methods are effective at removing target objects. PlanGen slightly outperforms CyCLeGen in terms of SR and IS, while CyCLeGen achieves comparable LocalCLIP scores. This observation suggests that for well-constrained editing tasks such as object removal, where the target region and objective are explicit, the benefit of cycle-based optimization is less pronounced, though it does not degrade deletion fidelity or image realism.

Overall, these results demonstrate that CyCLeGen provides more stable and reliable performance for complex image editing tasks requiring precise semantic alignment, while maintaining competitive object removal capability and visual quality.

A.6 Reward Curve

Fig.5 shows the reward curves during GRPO interleave training. For the generation task, both the HPSv2 aesthetic score and Layout IoU exhibit steady improvement throughout training, indicating that the model progressively learns to generate images with better visual quality and more accurate spatial layouts. For the understanding task, the IoU reward and CLIP image similarity both increase consistently, demonstrating enhanced visual grounding and improved cycle consistency between generation and understanding. All four reward signals plateau after approximately 1000 steps, suggesting that the interleave training converges efficiently within this regime.

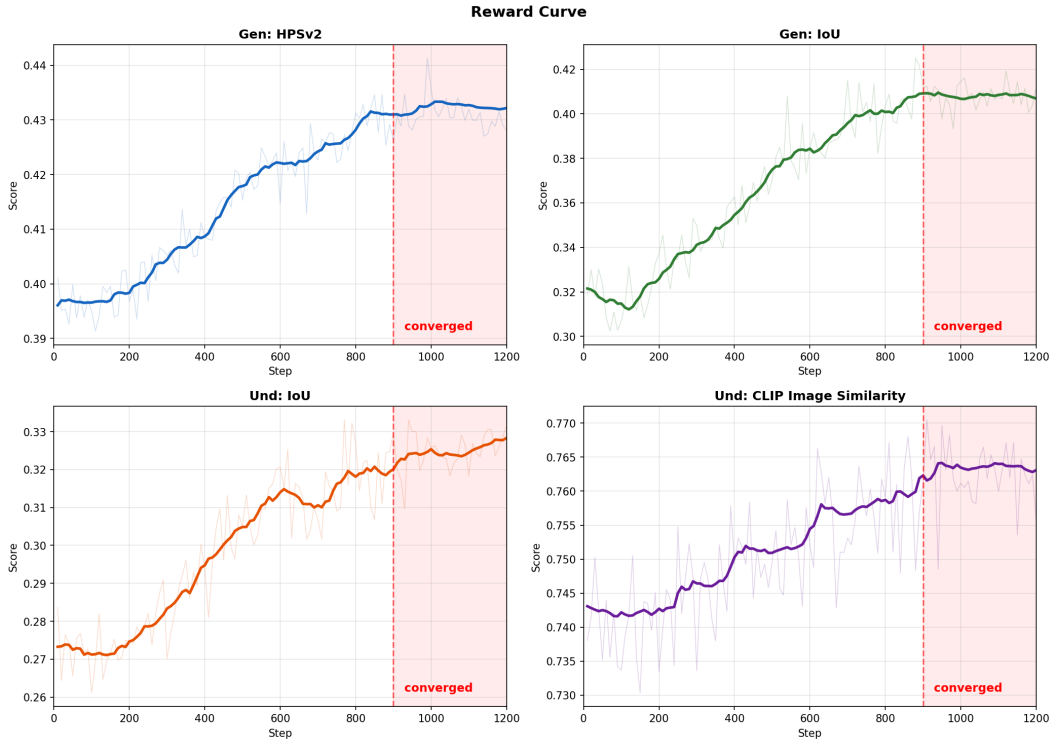

Figure 5: Reward curves during GRPO interleave training.

B Limitations

Although our unified cycle-consistent framework demonstrates strong improvements in both layout understanding and layout-conditioned generation, three limitations remain.

Model capacity constraints. Our current implementation is built on a 1.5B-parameter Plangen[1] backbone, which is significantly smaller than contemporary vision–language foundation models. While this lightweight design enables efficient training and experimentation, it also limits the model's ability to perform VQA and open-ended reasoning tasks.

Quality of RL data. Our reinforcement learning stage relies on data from open-source dataset LayoutSAM[9], which is labeled by Grounding-Dino[39], which may contain structural inaccuracies, visual artifacts, or alignment biases. Although our GRPO procedure improves robustness, the learning signal is ultimately constrained by the quality of these synthetic samples. Improving the fidelity of RL data, either through stronger generative models, human preference signals, or automated data filtering—represents an important direction for enhancing overall performance.

Generation diversity Cycle consistency may be less effective in scenes with dense or ambiguous layouts, where strong structural constraints can limit generation diversity.

Despite these limitations, our model is a universal framework that can be applied to different base models. Our results also highlight the promise of cycle-consistent unified modeling, and we believe that addressing model capacity and data quality will further unlock the full potential of our framework.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract and introduction accurately reflect the core contribution (cycle-consistent coupling of understanding and generation via CycleGRPO) and its demonstrated improvements across tasks.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes],

   Justification: The paper explicitly discusses limitations including model capacity constraints, RL data quality, and reduced generation diversity under complex layouts in appendix.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

Answer: [Yes],

Justification: The paper provides the necessary assumptions and a complete, standard GRPO-based formulation, though it builds on prior work rather than deriving new proofs from scratch.

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 paper provides key details such as datasets, training setup, hyperparameters, and reward design.

Guidelines:

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

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

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

   Answer: [Yes],

   Justification: The datasets used are publicly available and the code will be provided 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: The paper specifies key training details such as data source, splits, hyperparameters (e.g., rollout size, reward weights), and training procedure.

   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: The paper reports point estimates of metrics without error bars or statistical significance analysis.

   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 paper specifies the GPU type (8×A6000), training time (40 GPU hours), and some setup details.

   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 paper presents standard machine learning research with no apparent ethical concerns or violations of the NeurIPS Code of Ethics, and does not involve sensitive data or harmful applications.

   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: [N/A].

Justification: The paper focuses on a general-purpose modeling and training framework without targeting specific real-world deployment scenarios or sensitive domains; therefore, it does not introduce new societal risks beyond those already associated with existing vision-language models.

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: The paper does not release new datasets or deploy a high-risk model, but rather proposes a training framework built on existing models, so no specific safeguards are required.

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: The paper properly cites prior work, datasets, and models, indicating appropriate credit.

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: [N/A].

    Justification: The paper does not introduce new public assets.

    Guidelines:

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

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

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

    Answer: [N/A].

    Justification: The paper does not involve crowdsourcing experiments or research with human subjects.

    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: The paper does not involve human subjects, so there are no participant risks or IRB considerations.

Guidelines:

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

16. **Declaration of LLM usage**

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

Answer: [Yes],

Justification: The paper uses pretrained vision-language models as the backbone.

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.