InstanceAssemble: Layout-Aware Image Generation via Instance Assembling Attention

Qiang Xiang[1,2], **Shuang Sun**[2], **Binglei Li**[1,3],
Dejia Song[2], **Huaxia Li**[2], **Yibo Chen**[2], **Xu Tang**[2], **Yao Hu**[2], **Junping Zhang**[1*]
[1]Shanghai Key Laboratory of Intelligent Information Processing,
College of Computer Science and Artificial Intelligence, Fudan University
[2]Xiaohongshu Inc. [3]Shanghai Innovation Institute
{qxiang24, blli24}@m.fudan.edu.cn, jpzhang@fudan.edu.cn,
{sunshuang1, dejiasong, lihuaxia, zhaohaibo, tangshen, xiahou}@xiaohongshu.com

Figure 1: **Layout-aware image generation result by InstanceAssemble**. We show image generation result under precise layout control, ranging from simple to intricate, sparse to dense layouts.

Abstract

Diffusion models have demonstrated remarkable capabilities in generating high-quality images. Recent advancements in Layout-to-Image (L2I) generation have leveraged positional conditions and textual descriptions to facilitate precise and controllable image synthesis. Despite overall progress, current L2I methods still exhibit suboptimal performance. Therefore, we propose InstanceAssemble, a novel architecture that incorporates layout conditions via instance-assembling attention, enabling position control with bounding boxes (bbox) and multimodal content control including texts and additional visual content. Our method achieves flexible adaption to existing DiT-based T2I models through light-weighted LoRA modules. Additionally, we propose a Layout-to-Image benchmark, Denselayout, a comprehensive benchmark for layout-to-image generation, containing 5k images with 90k instances in total. We further introduce Layout Grounding Score (LGS), an interpretable evaluation metric to more precisely assess the accuracy of L2I generation. Experiments demonstrate that our InstanceAssemble method achieves state-of-the-art performance under complex layout conditions, while exhibiting strong compatibility with diverse style LoRA modules. The code and pretrained models are publicly available at `https://github.com/FireRedTeam/InstanceAssemble`.

*Corresponding author.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

1 Introduction

Diffusion models [22] have revolutionized image generation task, with architectures like Diffusion Transformer (DiT) [40] offering superior quality over traditional UNet-based frameworks. Recent implementations such as Stable Diffusion 3/3.5 [15, 49] and Flux.1 [4] further enhance text-to-image alignment, paving the way for advancements in layout-controlled generation. Layout-to-Image (L2I) generation is a task that focuses on creating images under layout conditions, allowing users to define spatial positions and semantic content of each instance explicitly. This task faces several significant challenges: (***i***) ensuring precise layout alignment while maintaining high image quality, (***ii***) preserving object positions and semantic attributes accurately during the iterative denoising process of diffusion models, and (***iii***) supporting various types of reference conditions, such as texts, images and structure information. These challenges highlight the complexity of achieving robust and flexible layout-controlled image generation.

Existing L2I methods can be broadly categorized into training-free and training-based approaches, both possessing distinct advantages and limitations. Training-free methods [55, 7, 13, 8, 5, 27] rely on heuristic techniques without modifying the base model. However, these methods often exhibit degraded performance in complex layouts, demonstrate high sensitivity to hyperparameter tuning, and suffer from slow inference speed, which make them less practical for real-world applications. In contrast, training-based methods [63, 53, 64, 29, 61] involve training specific layout modules to improve layout alignment, which introduces a significant amount of extra parameters and increases training complexity and resource requirements. Additionally, existing L2I evaluation metrics exhibit inaccuracies, such as false acceptance and localization errors. These identified shortcomings necessitate algorithm innovation for effective and efficient layout-controlled image generation.

Therefore, we propose **InstanceAssemble**, a novel framework that systematically tackles these issues through innovative design and efficient implementation. Our approach introduces a cascaded InstanceAssemble structure, which employs a multimodal interaction paradigm to process global prompts and instance-wise layout conditions sequentially. By leveraging the Assemble-MMDiT architecture, we apply an independent attention mechanism to the semantic content of each instance, thus enabling effective handling of dense and complex layouts. Furthermore, we adopt LoRA [23] for lightweight adaptation, adding only 71M parameters to SD3-Medium (2B) and 102M to Flux.1 (11.8B). Our method enables position control with bounding boxes and multimodal content control including texts and additional visual content. This lightweight design preserves the capabilities of the base model while enhancing flexibility and efficiency. We also introduce a novel metric called **Layout Grounding Score (LGS)** to ensure accurate evaluation for L2I generation, alongside a test dataset **DenseLayout**. This metric provides a consistent benchmark for assessing layout alignment.

Our method achieves state-of-the-art performance across benchmarks and demonstrates robust layout alignment under a wide variety of scenarios, ranging from simple to intricate, sparse to dense layouts. Notably, despite being trained on sparse layouts ($\leq$ 10 instances), our approach maintains robust generalization capability on dense layouts ($\geq$ 10 instances), confirming the effectiveness of our proposed InstanceAssemble. The main contributions are listed below.

1. We propose a cascaded InstanceAssemble structure that processes global text prompts and layout conditions sequentially, enabling robust handling of complex layouts through an independent attention mechanism.

2. By leveraging LoRA [23], we achieve efficient adaptation with minimal extra parameters (3.46% on SD3-Medium and 0.84% on Flux.1), supporting position control with multimodal content control while preserving capabilities of base model.

3. We propose a new test dataset DenseLayout and a novel metric Layout Grounding Score (LGS) for Layout-to-Image evaluation. Experimental results demonstrate that our approach achieves state-of-the-art performance and robust capability under complex and dense layout conditions.

2 Related Work

Text-to-Image Generation Text-to-image synthesis [47, 45, 42, 40, 30, 17, 6, 1, 44, 3] has witnessed rapid progress with the development of diffusion models. Initial works [44, 3, 47, 45] utilize UNet [46] as the denoising architecture, leveraging cross-attention mechanisms to inject text-conditioning

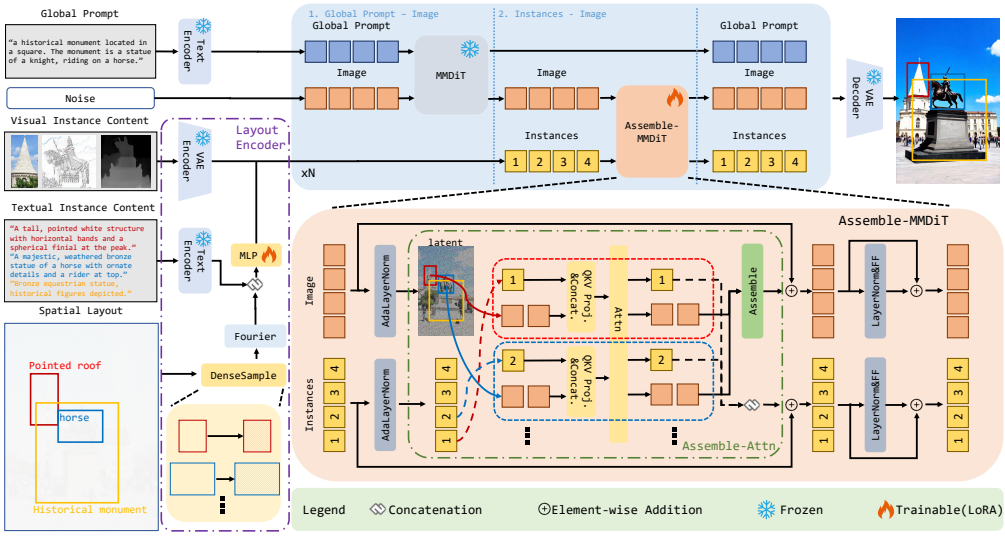


Figure 2: **The proposed InstanceAssemble pipeline.** Various layout conditions are processed by the Layout Encoder to obtain instance tokens, which guide the image generation via Assemble-MMDiT. In Assemble-MMDiT, the instance tokens interact with image tokens through the Assembling-Attn.

signals. Recently, researches [6, 15, 49, 4, 34] have used the Multimodal Diffusion Transformer (MMDiT) architecture, marking a significant improvement.

Layout-to-Image Generation Layout-to-Image generation enables image generation under layout conditions, which is defined as spatial positions with textual descriptions. Existing approaches can be broadly categorized into training-free and training-based paradigms.

Training-free methods leverage pretrained text-to-image diffusion models without additional training. A common strategy involves gradient-based guidance during denoising to align with layout conditions [55, 12, 41, 13, 18, 50]. Also, there are methods that directly manipulate latents through well-defined replacing or merging operations [8, 48, 2] or enforce layout alignment via spatially constrained attention masks [5, 20]. GrounDiT [27] exploits semantic sharing in DiT: a cropped noisy patch and the full image become semantic clones when denoised together, enabling layout-to-image generation by jointly denoising instance regions with their corresponding image context. Other approaches generate each instance separately and employ inpainting techniques to compose the final image [37]. However, these methods demonstrate decent performance primarily on simple and sparse layouts, while their accuracy decreases in more complex layouts. Some methods require hyperparameter tuning specific to different layout conditions, reducing their adaptability. Furthermore, additional gradient computations or latent manipulations result in slow inference speed, thus limiting their applicability in real-world scenarios.

Training-based methods explicitly incorporate layout conditioning through architectural modifications. Most approaches inject spatial constraints via cross-attention [63, 57, 36, 25, 59, 16, 54] or self-attention [10, 53, 28]. Some works propose dedicated layout encoding modules [9, 64, 65, 58, 62] or adopt a two-stage pipeline that generates images after predicting a depth map with layout conditions [14, 66]. Other works leverage autoregressive image generation models [19]. These methods suffer high computational costs due to excessive parameters.

3 Method

Preliminaries Recent state-of-the-art text-to-image models such as SD3 [15] and Flux [4] adopt the Multimodal Diffusion Transformer (MMDiT) as the backbone for generation. Unlike traditional UNet-based cross-attention approaches, MMDiTs treat image and text modalities in a symmetric manner, which leads to stronger prompt alignment and controllability. These models are trained under the flow matching framework [33], which formulates generation as learning a continuous velocity field that transports noise to data. Given a clean latent $\mathbf{x}$ and Gaussian noise $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$, an

interpolated latent is defined as

$$\mathbf{z}_t = (1-t)\mathbf{x} + t\boldsymbol{\epsilon}, \quad t \in [0,1]. \tag{1}$$

The training objective minimizes the squared error between the predicted velocity and the target velocity $(\boldsymbol{\epsilon} - \mathbf{x})$:

$$\mathcal{L}_{\text{FM}} = \mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbf{I}),\,\mathbf{x},\,t}\left[\left\|v_\theta(\mathbf{z}_t, t, \mathbf{y}) - (\boldsymbol{\epsilon} - \mathbf{x})\right\|_2^2\right], \tag{2}$$

where v_θ is implemented with an MMDiT backbone.

Problem Definition Layout-to-Image generation aims to synthesize images with precise control through a global prompt $\boldsymbol{p}$ and instance-wise layout conditions $\boldsymbol{L}$. The layout conditions comprise N instances $\{l_i\}_{i=1}^N$, where each instance l_i is defined by its spatial position b_i and content c_i:

$$\boldsymbol{L} = \{l_1, \ldots, l_N\}, \quad \text{where } l_i = (c_i, b_i). \tag{3}$$

In our framework, spatial positions are represented as bounding boxes, while instance content can be specified through multiple modalities: textual instance content and additional visual instance content, including reference images, depth maps and edge maps.

We propose InstanceAssemble, a framework with a **Layout Encoder** to encode the layout conditions and **Assemble-MMDiT** to effectively integrate the encoded layout conditions with image features.

3.1 Layout Encoder

We use a Layout Encoder (Fig. 2 left-bottom panel) to encode each instance l_i, and the tokens are denoted as $\boldsymbol{h}^{\boldsymbol{L}} = [h^{l_1}, \ldots, h^{l_N}]$ which represents the layout information of each instance. Given the spatial position of the instance (bounding box), we first enhance the spatial representation through **DenseSample**. Given a bounding box $b_i = (x_1, y_1, w, h) \in [0,1]^4$ with top-left coordinates (x_1, y_1) and size (w, h), we generate K^2 uniformly spaced points:

$$\mathcal{P}_i = \left\{ \left(x_1 + k_x \cdot \frac{w}{K}, y_1 + k_y \cdot \frac{h}{K} \right) \middle| k_x, k_y \in \{0, \ldots, K-1\} \right\} \tag{4}$$

Then, following GLIGEN[28], we compute the textual instance tokens as:

$$h_l^i = \text{MLP}\left([\boldsymbol{\tau}(c_i), \text{Fourier}(\mathcal{P}_i)]\right), \tag{5}$$

where $\boldsymbol{\tau}$ represents the text encoder, $\text{Fourier}(\cdot)$ denotes Fourier embedding [51], $[\cdot,\,\cdot]$ denotes concatenation along the feature dimension, and MLP is a multi-layer perception.

Additionally, we can use additional visual instance content to better improve performance. Given the visual instance content, we first extract features using the VAE encoder of the base model, then project them to the unified instance token space through a MLP:

$$h_l^i = \text{MLP}\left(\text{VAE}(c_i)\right). \tag{6}$$

3.2 Assemble-MMDiT

We observe that applying attention between all image tokens and instance tokens results in suboptimal performance under complex layout conditions (e.g., overlapping, tiny objects). To address this, we introduce **Assemble-MMDiT** (Fig. 2, right-bottom panel), which enhances the location of each instance while maintaining compositional coherence with other instances. Our method processes each instance independently through attention modules with its associated image tokens, followed by weighted feature assembling.

Formally, given image tokens $\boldsymbol{h} \in \mathbb{R}^{C\times W\times H}$ (where C denotes the latent channel size and $[W, H]$ the latent size) and instance tokens $\boldsymbol{h}^l \in \mathbb{R}^{C\times N}$, we apply AdaLayerNorm [56], followed by our proposed **Assembling-Attn**, as shown in Fig. 2 (right-bottom panel). We crop the image tokens $\boldsymbol{h}^z$ by the bbox $\boldsymbol{b}_i$ of each instance and get $\boldsymbol{h}_{l_i}^z = \boldsymbol{h}^z[\boldsymbol{b}_i] \in \mathbb{R}^{C\times w\times h}$. Then, we project the cropped image tokens $\boldsymbol{h}_{l_i}^z$ and their corresponding instance tokens l_i into queries $(Q^{z_{l_i}}, Q^{l_i})$, keys $(K^{z_{l_i}}, K^{l_i})$, and values $(V^{z_{l_i}}, V^{l_i})$, and then apply attention:

$$\boldsymbol{h}_{l_i}^{z'}, \boldsymbol{h}^{l_i'} = \text{Attention}\left([Q^{z_{l_i}}, Q^{l_i}], [K^{z_{l_i}}, K^{l_i}], [V^{z_{l_i}}, V^{l_i}]\right). \tag{7}$$

where $[\cdot, \cdot]$ denotes concatenation along the token dimension. The updated tokens are assembled across instances. Let $M \in \mathbb{N}^{W \times H}$ represent the instance density map, calculating the counts of instances. The assembled image tokens $\boldsymbol{h}^{z'}$ and instance tokens $\boldsymbol{h}^{l'}$ are computed as:

$$\begin{aligned} \boldsymbol{h}^{z'} : \boldsymbol{h}^{z'}[:, i, j] &= \frac{1}{M[i,j]} \sum_{k=1}^{N} \boldsymbol{h}_{l_k}^{z'}[:, i, j], \text{ where } i \in [0, W-1], j \in [0, H-1] \\ \boldsymbol{h}^{l'} : \boldsymbol{h}^{l'}[:, k] &= \boldsymbol{h}^{l'_k}. \end{aligned} \tag{8}$$

As illustrated in Fig. 3, the top row demonstrates that our assembling mechanism ensures instance tokens attend only to relevant image regions, where unrelated regions are left in black. The middle row reveals that the mechanism effectively guides global prompt tokens to focus on their correct spatial positions. In contrast, generation without explicit layout control (bottom row) results in localization errors ("British Shorthair" in wrong location) or semantic inconsistencies ("dog" missing).

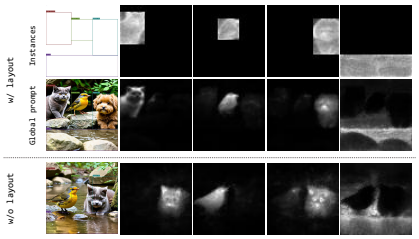


Figure 3: (Top) instance-image attention map w/ layout. (Middle) global prompt-image attention map w/ layout. (Bottom) global prompt-image attention map w/o layout.

Furthermore, to preserve the generation capability of the original model and mitigate conditional conflicts between global prompt and layout conditions, we employ a cascaded mechanism as shown in Fig. 2 (right-above panel). In our design, the global text prompt and image latents are passed through original MMDiT first, then the image tokens along with instance tokens are processed by our Assemble-MMDiT module. The first step captures global context and ensures generation quality, while the second step ensures instance layout alignment. Besides, we train Assemble-MMDiT with LoRA, significantly reducing both the training cost and inference costs.

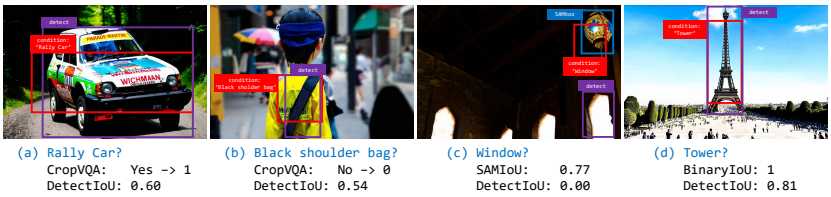


Figure 4: **Failure cases of other metrics.** (**a**) false acceptance in CropVQA,(**b**) false rejection in CropVQA, (**c**) localization error in SAMIoU, and (**d**) discontinuous in BinaryIoU.

3.3 Benchmark: DenseLayout and Layout Grounding Score

The Layout-to-Image task aims to generate images that align precisely with provided layouts, evaluating both spatial accuracy and semantic consistency (e.g., color, texture, and shape, if provided). The existing metrics (AP/AR) for object detection [10, 28, 55, 53] are suboptimal. They assume a fixed category set and rely on inappropriate precision/recall for binary layout outcomes. VLM-based cropped VQA methods [61] suffer false acceptance (Fig. 4 (a)) and false rejection (Fig. 4 (b)). While spatial-only metrics like SAMIoU [11] ignore appearance consistency(Fig. 4 (c)), GroundingDINO-based [35] binary IoU thresholds [64, 54] fail to capture continuous layout precision(Fig. 4 (d)). Thus, we propose **Layout Grounding Score (LGS)**, which integrates both spatial accuracy and semantic accuracy:

1. **Spatial Accuracy (DetectIoU)**: we detect all instances via an off-the-shelf detector [35], compute the IoU against condition bbox, and report the global mean IoU across all instances for equal weighting.

2. **Semantic Accuracy**: for instances with IoU>0.5, we crop the predicted region and assess the semantic accuracy by its attribute consistency (color, texture, shape) via VLM-based VQA [60].

LGS supports open-set evaluation, uses DetectIoU to evaluate spatial accuracy and decouples the spatial and semantic check to avoid CropVQA [61] failures (shown in Fig. 4). Furthermore, we

Table 1: **Quantitative comparison between our SD3-based InstanceAssemble and other L2I methods on LayoutSAM-Eval.** ⋆ The CropVQA score is proposed in Creatilayout [61] and the score of InstanceDiff, MIGC and CreatiLayout is borrowed from CreatiLayout [61].

LayoutSAM-Eval	CropVQA⋆				Layout Grounding Score				Global Quality		
	spatial↑	color↑	texture↑	shape↑	mIoU↑	color↑	texture↑	shape↑	VQA↑	Pick↑	CLIP↑
Real Images(Upper Bound)	98.95	98.45	98.90	98.80	88.85	88.07	88.71	88.62			
InstanceDiff (SD1.5)	87.99	69.16	72.78	71.08	78.16	63.14	**66.82**	**65.86**	86.42	21.16	11.73
MIGC (SD1.4)	85.66	66.97	71.24	69.06	62.87	50.70	52.99	51.77	88.97	20.69	12.56
HICO (realisticVisionV51)	90.92	69.82	73.25	71.69	70.68	53.16	55.71	54.61	86.53	21.77	9.47
CreatiLayout (SD3-M)	92.67	74.45	77.21	75.93	45.82	38.44	39.68	39.24	92.74	21.71	**13.82**
InstanceAssemble(ours) (SD3-M)	**94.97**	**77.53**	**80.72**	**80.11**	**78.88**	**63.89**	66.27	**65.86**	**93.12**	**21.79**	12.76

Table 2: **Quantitative comparison between our InstanceAssemble and other L2I methods on DenseLayout.**

DenseLayout	Layout Grounding Score				Global Quality		
	mIoU↑	color↑	texture↑	shape↑	VQA↑	Pick↑	CLIP↑
Real Images(Upper Bound)	92.35	76.52	80.78	79.78			
InstanceDiff (SD1.5)	47.31	29.48	33.36	32.43	88.79	20.87	11.73
MIGC (SD1.4)	34.39	22.10	23.99	23.45	91.18	20.74	12.81
HICO (realisticVisionV51)	22.42	10.52	11.69	11.46	74.42	20.51	8.16
CreatiLayout (SD3-Medium)	15.54	11.69	12.34	12.17	93.42	**21.88**	**12.89**
InstanceAssemble(ours) (SD3-Medium)	**52.07**	**33.77**	**36.21**	**35.81**	**93.54**	21.68	12.58
Regional-Flux (Flux.1-Dev)	14.06	11.34	11.91	11.84	92.94	**22.67**	10.66
RAG (Flux.1-Dev)	17.23	14.22	14.62	14.55	92.16	22.28	11.01
InstanceAssemble(ours) (Flux.1-Dev)	43.42	27.60	29.50	29.14	93.36	21.98	**11.38**
InstanceAssemble(ours) (Flux.1-Schnell)	**45.33**	**27.73**	**30.06**	**29.62**	**93.52**	21.72	10.78

introduce DenseLayout, a dense evaluation dataset for L2I, which consists of 5k images with 90k instances (18.1 per image). The images in DenseLayout are generated by Flux.1-Dev, tagged by RAM++ [24], detected by GroundingDINO [35], recaptioned by Qwen2.5-VL [43], and filtered to retain those with $\geq$15 instances, thus providing dense layout conditions.

3.4 Training and Inference

During training, we freeze the parameters of the base model and only update the proposed Layout Encoder and Assemble-MMDiT module. We denote the adding parameters by θ'. The training objective is given by

$$\mathcal{L} = \mathbb{E}_{\epsilon\sim\mathcal{N}(0,\mathbf{I}),\,\mathbf{x},\,t,\,\mathbf{p},\boldsymbol{L}}\left[\left\|v_{\{\theta,\theta'\}}\big(\mathbf{z}_t, t, \mathbf{p}, \boldsymbol{L}\big) - (\epsilon - \mathbf{x})\right\|_2^2\right], \tag{9}$$

where $\mathbf{z}_t = (1-t)\mathbf{x} + t\epsilon$. During inference, layout-conditioned denoising is applied during the first 30% of diffusion steps, as the layout primarily forms in early stages [28, 64].

4 Experiments

4.1 Experimental Setup

Implementation Details The textual-only InstanceAssemble is trained on SD3-Medium [15] and Flux.1-Dev [4] and the version with additional visual instance content is only trained on SD3-Medium. We freeze the pretrained MMDiT backbone and only adapt the Layout Encoder and LoRA modules of Assemble-MMDiT. Assemble-MMDiT is initialized from pretrained weights, and LoRA with rank=4 is applied. In the SD3-based model all Assemble-MMDiT blocks are adapted, while in the Flux-based model we adapt eight blocks (seven double-blocks and one single block) due to resource constraints. During inference, the LoRA-based Assemble-MMDiT is activated for the first 30% of denoising steps, while the global prompt–image phase uses the frozen backbone. This design yields 71M (SD3-M) and 102M (Flux.1-Dev) additional parameters for the textual-only setting; the variant with additional visual instance content (SD3-M) introduces 85M parameters. All models are trained on LayoutSAM [61] at 1024×1024 with Prodigy, for 380K iterations (batch size 2) on SD3-M and

Table 4: **Parameter addition and time efficiency under sparse and dense layout conditions.** We evaluated on 10% of the LayoutSAM-Eval and DenseLayout datasets at 1024×1024 resolution. * are optimized for 512×512 resolution, so their results are reported at this scale.

	Parameter Addition (relative parameter addition(%))	Time Efficiency(s) (relative runtime increase(%)) Sparse Layout	Dense Layout
InstanceDiff* (SD1.5)	369M (43%)	14.37 (+771%)	44.81 (+2754%)
MIGC(SD1.4)	57M (6.64%)	14.41 (+25.4%)	21.58 (+87.5%)
HICO* (realisticVisionV51)	361M(33.9%)	4.11 (+92.9%)	9.93 (+320%)
CreatiLayout(SD3-M)	1.2B (64.0%)	4.37 (+14.4%)	4.42 (**+14.8%**)
Regional-Flux(Flux.1-Dev)	-	15.29 (+113%)	37.47 (+418%)
RAG(Flux.1-Dev)	-	15.69 (+119%)	21.14 (+192%)
InstanceAssemble(ours)(SD3-M)	71M (3.46%)	7.19 (+88.2%)	13.38 (+248%)
InstanceAssemble(ours)(Flux.1-Dev)	**102M (0.84%)**	8.21 (+14.3%)	10.28 (+41.9%)
InstanceAssemble(ours)(Flux.1-Schnell)	**102M (0.84%)**	**1.41 (+8.46%)**	**1.70** (+28.8%)

300K iterations (batch size 1) on Flux.1-Dev, using 8×H800 GPUs (7 days for SD3-M; 5 days for Flux.1-Dev).

Evaluation Dataset We use LayoutSAM-Eval [61] to evaluate performance on fine-grained open-set sparse L2I dataset, containing 5k images and 19k instances in total (3.8 instances per image). To assess performance on fine-grained open-set dense L2I evaluation dataset, we use the proposed *DenseLayout*, which consists of 5k images and 90k instances in total (18.1 instances per image). Following conventional practice, we also evaluate on coarse-grained close-set L2I evaluation dataset COCO [31]. We combine COCO-Stuff and COCO-Instance annotations to create our COCO-Layout evaluation dataset, containing 5k images and 57k instances in total (11.5 instances per image).

Evaluation Metric We evaluate the accuracy of L2I generation using our proposed LGS metric along with CropVQA proposed by CreatiLayout [61], measuring spatial and semantic accuracy. We also employ multiple established metrics to measure overall image quality and global prompt alignment, including VQA Score [32], PickScore [26] and CLIPScore [21].

4.2 Evaluation on L2I with Textual-Only Content

Fine-Grained Open-Set Sparse L2I Generation Tab. 1 presents the quantitative results of InstanceAssemble on LayoutSAM-Eval [61], reporting results using our proposed LGS, CropVQA [61] and global quality metrics. Our proposed InstanceAssemble not only achieves SOTA in spatial and semantic accuracy of each instance, but also demonstrates superior global quality.

Fine-Grained Open-Set Dense L2I Generation Tab. 2 presents results on DenseLayout. With the same SD3-Medium backbone, InstanceAssemble significantly outperforms CreatiLayout (mIoU: 52.07 vs. 15.54) while maintaining comparable global quality. On Flux.1, it also yields large gains over Regional-Flux and RAG (e.g., mIoU: 43.42 vs. 17.23 for RAG), showing that our cascaded Assemble-Attn design generalizes well across backbones. Compared to earlier UNet-based approaches such as InstanceDiff and MIGC, our method achieves higher spatial and semantic accuracy (mIoU: 52.07 vs. 47.31) without sacrificing realism. Overall, InstanceAssemble establishes consistent improvements in layout alignment while ensuring high image quality under challenging dense layouts.

Table 3: **Comparison between our SD3-based InstanceAssemble and other L2I methods on COCO-Layout.** Since COCO don"t have detailed description for each instance, we cannot evaluate the attribute accuracy and only report the spatial accuracy - mIoU.

COCO-Layout	LGS mIoU↑	Global Quality VQA↑	Pick↑	CLIP↑
Real Images(Upper Bound)	49.14			
InstanceDiff (SD1.5)	**30.39**	75.77	20.75	24.41
MIGC (SD1.4)	27.36	70.32	20.20	23.58
HICO (realisticVisionV51)	18.88	50.61	20.38	20.72
CreatiLayout (SD3-M)	7.12	87.79	21.22	25.59
InstanceAssemble(ours) (SD3-M)	27.85	**89.06**	**21.58**	**25.68**

Coarse-Grained Closed-Set L2I Generation Tab. 3 presents the quantitative result of InstanceAssemble on COCO-Layout. Our proposed InstanceAssemble surpasses previous methods in overall image quality but lags slightly behind InstanceDiff [53] in layout precision. We attribute this gap to:

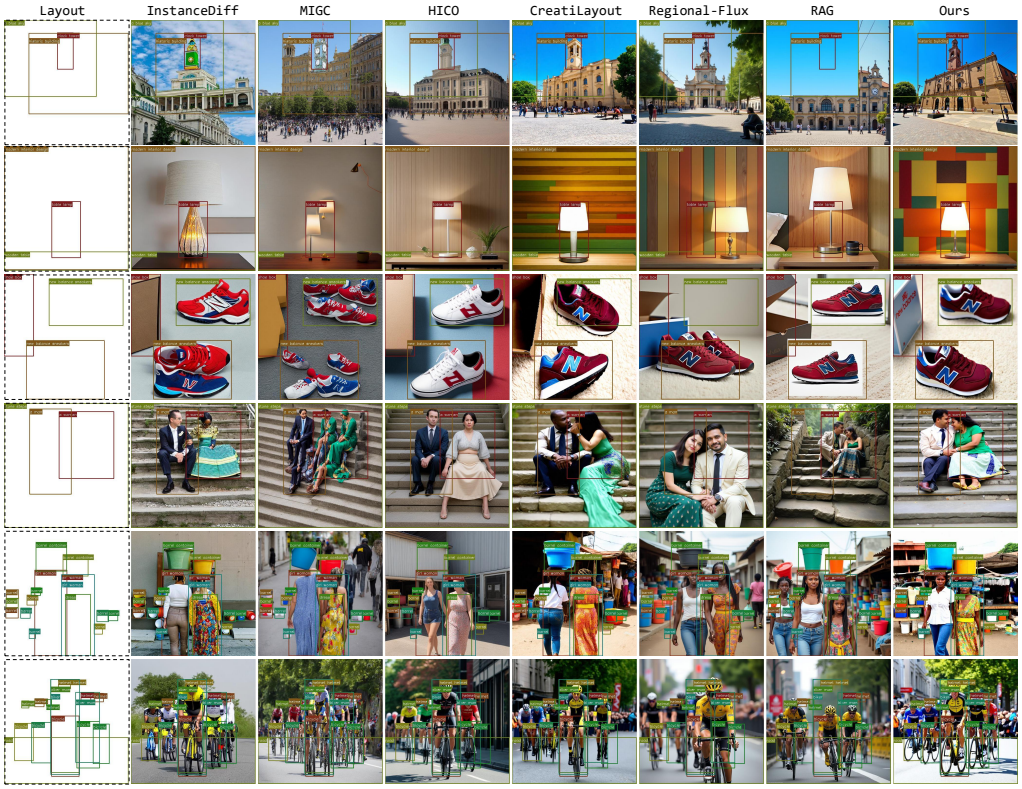


Figure 5: **Qualitative comparison of InstanceAssemble with other methods.**

(i) InstanceDiff's fine-grained COCO training data with per-entity attribute annotations, and (ii) its entity-wise generation strategy, which improves precision at significant computational cost (Tab. 4).

Qualitative Comparison The comparative results in Fig. 5 demonstrate that our proposed InstanceAssemble method achieves superior spatial precision and instance-caption alignment compared to baseline methods. For example, in the third row, both InstanceDiff [53] and MIGC [64] generate more than one shoes; HICO [9] fails to generate the specified NewBalance shoe; Regional-Flux [5] does not adhere to the layout conditions; and the shoe generated by RAG [8] is not properly fused with the background. In contrast, our method generates the correct instance, accurately placed and seamlessly integrated with the scene.

Time Efficiency and Parameter Addition We compare time efficiency and parameter addition with other L2I methods, as shown in Tab. 4. Our method achieves SOTA performance on layout alignment with acceptable time efficiency and minimal parameter addition.

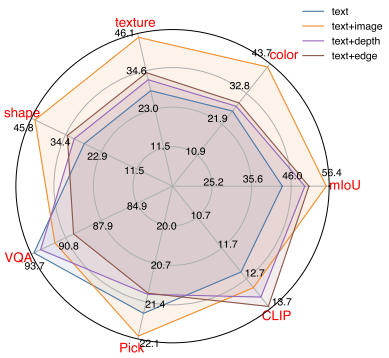


Figure 6: **Quantitative results of InstanceAssemble with additional visual instance content.**

4.3 Evaluation on L2I with Additional Visual Content

We evaluate three additional visual instance content: image, depth, and edge (see Fig. 6). Unsurprisingly, using image as additional instance content yields the best performance, as it provides rich visual information. Although depth and edge capture texture and shape features, their performance remains inferior compared to image instance content. Nevertheless, visual modalities outperform textual-only instance

Table 5: **Ablation study on our proposed components on DenseLayout.** "Assemble" refers to the presence of the Assemble-Attn module (architectural design). "Cascaded" indicates the interaction order: (✔) means global prompt–image interaction followed by instance–image interaction (cascaded structure), while (✘) means both are applied in parallel. "LoRA" specifies the training strategy for the Assemble-MMDiT module: (✔) indicates training with LoRA, while (✘) indicates full fine-tuning. "DenseSample" denotes whether the DenseSample spatial encoding is used.

Assemble	Cascaded	LoRA	DenseSample	Layout Grounding Score				
				mIoU↑	color↑	texture↑	shape↑	VQA↑
✘	✘	✘	✘	11.69	9.16	9.68	9.56	93.75
✔	✘	✘	✘	43.98	24.19	26.95	26.75	84.57
✔	✔	✘	✘	45.96	29.61	31.50	31.09	92.71
✔	✔	✔	✘	51.28	32.68	34.94	34.58	93.33
✔	✔	✔	✔	**52.07**	**33.77**	**36.21**	**35.81**	**93.54**

content. Qualitative comparisons (Fig. 7) further demonstrate that visual instance content leads to superior texture and shape alignment compared to text.

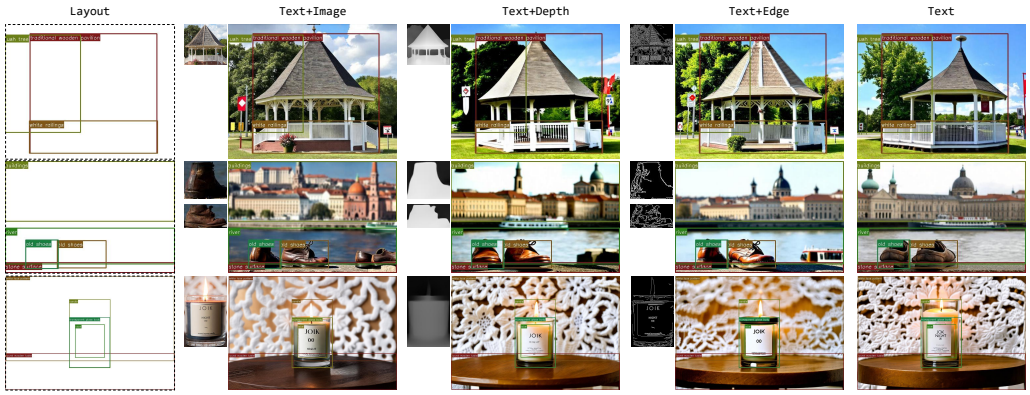


Figure 7: **Qualitative results of InstanceAssemble with additional visual instance content.**

4.4 Ablation Study

We evaluated the contribution of each proposed component on SD3-M based InstanceAssemble in Tab. 5. The **base model** (SD3-Medium without additional modules) yields a very low Layout Grounding Score, indicating poor layout and content control when instance information is not explicitly modeled. The introduction of Assemble-Attn module elevates spatial accuracy (mIoU to 43.98) and boosts semantic metrics (color/texture/shape to 24.19/26.95/26.75). The cascaded design (✔: prompt–image followed by instance–image; ✘: parallel) resolves global quality degradation while maintaining layout alignment. Using **LoRA** to train Assemble-MMDiT improves performance for two reasons: (1) it retains the base model's capabilities compared to the fully fine-tuned version, and (2) it enables effective layout control with far fewer trainable parameters by introducing only lightweight low-rank matrices on attention projections. Finally, **DenseSample** further enhances spatial accuracy, instance semantic accuracy and image quality. Together, these refinements progressively collectively optimize layout to image modeling without compromising generation ability.

4.5 Applications

We demonstrate that InstanceAssemble is versatile and applicable to various tasks. It seamlessly integrates with domain-specific LoRA modules for multi-domain style transfer while maintaining layout consistency, as shown in Fig. 8. Our proposed InstanceAssemble can cooperate with distilled models such as Flux.1-Schnell [4], as illustrated in Tab. 2, achieving geometric layout control and detailed synthesis. Our approach demonstrates both style adaptability and computational efficiency, making it well-suited for controllable generative design applications.

Figure 8: **The adaption of Cute3D [52]/ Oil Painting [39]/ Ghibli [38] LoRA with our methods**. The proposed InstanceAssemble successfully adapts diverse style lora and maintaining superior layout alignment.

5 Conclusion

We present InstanceAssemble, a novel approach for Layout-to-Image generation. Our method achieves state-of-the-art layout alignment while maintaining high-quality generation capabilities of DiT-based architectures. We validate InstanceAssemble across textual instance content and additional visual instance content, demonstrating its versatility and robustness. Our layout control scheme also successfully adapts diverse style LoRAs while maintaining superior layout alignment, demonstrating cross-domain generalization capability. Futhermore, we introduce Layout Grounding Score metric and a DenseLayout evaluation dataset to validate performance under complex layout conditions.

Limitations and Future Work While our work advances controllable generation by unifying precise layout control with the expressive power of diffusion models, several limitations remain. First, our design currently requires sequential Assemble-MMDiT calls, which may incur inefficiency; exploring parallelization strategies is an important direction. Second, although our approach is effective under a wide range of layouts, image fidelity can degrade in extremely dense or highly complex cases.

Broader Impacts InstanceAssemble expands the frontier of structured visual synthesis by providing fine-grained layout control and high-quality multimodal generation. However, its powerful generative capabilities may also introduce risks. In particular, malicious use could enable the creation of misleading or deceptive layouts, exacerbating the spread of disinformation. The model may also raise privacy concerns if applied to sensitive data, and like many generative systems, it inherits and may amplify societal biases present in training corpora. We encourage responsible deployment and continued investigation into safeguards that mitigate these risks while enabling beneficial applications in design, education, and accessibility.

Acknowledgments and Disclosure of Funding

This research was supported by the National Natural Science Foundation of China (NSFC 62576103, 62176059). The computations were conducted using the CFFF platform at Fudan University. Part of this work was carried out during an internship at Xiaohongshu.

References

[1] F. Bao, S. Nie, K. Xue, Y. Cao, C. Li, H. Su, and J. Zhu. All are Worth Words: A ViT Backbone for Diffusion Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22669–22679. IEEE, 2023.

[2] O. Bar-Tal, L. Yariv, Y. Lipman, and T. Dekel. MultiDiffusion: Fusing Diffusion Paths for Controlled Image Generation. In *Proceedings of the International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 1737–1752. PMLR, 2023.

[3] J. Betker, G. Goh, L. Jing, T. Brooks, J. Wang, L. Li, L. Ouyang, J. Zhuang, J. Lee, Y. Guo, et al. Improving image generation with better captions, 2023.

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

[5] A. Chen, J. Xu, W. Zheng, G. Dai, Y. Wang, R. Zhang, H. Wang, and S. Zhang. Training-free Regional Prompting for Diffusion Transformers, 2024. URL `https://arxiv.org/abs/2411.02395`.

[6] J. Chen, J. Yu, C. Ge, L. Yao, E. Xie, Z. Wang, J. T. Kwok, P. Luo, H. Lu, and Z. Li. PixArt-α: Fast Training of Diffusion Transformer for Photorealistic Text-to-Image Synthesis. In *Proceedings of the International Conference on Learning Representations*. OpenReview.net, 2024.

[7] M. Chen, I. Laina, and A. Vedaldi. Training-Free Layout Control with Cross-Attention Guidance. In *Winter Conference on Applications of Computer Vision*, pages 5331–5341. IEEE, 2024.

[8] Z. Chen, Y. Li, H. Wang, Z. Chen, Z. Jiang, J. Li, Q. Wang, J. Yang, and Y. Tai. Region-Aware Text-to-Image Generation via Hard Binding and Soft Refinement, 2024. URL `https://arxiv.org/abs/2411.06558`.

[9] B. Cheng, Y. Ma, L. Wu, S. Liu, A. Ma, X. Wu, D. Leng, and Y. Yin. HiCo: Hierarchical Controllable Diffusion Model for Layout-to-image Generation. In *Advances in Neural Information Processing Systems*, 2024.

[10] J. Cheng, X. Liang, X. Shi, T. He, T. Xiao, and M. Li. LayoutDiffuse: Adapting Foundational Diffusion Models for Layout-to-Image Generation, 2023. URL `https://arxiv.org/abs/2302.08908`.

[11] J. Cheng, Z. Zhao, T. He, T. Xiao, Z. Zhang, and Y. Zhou. Rethinking The Training And Evaluation of Rich-Context Layout-to-Image Generation. In *Advances in Neural Information Processing Systems*, 2024.

[12] G. Couairon, M. Careil, M. Cord, S. Lathuilière, and J. Verbeek. Zero-shot spatial layout conditioning for text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2174–2183. IEEE, 2023.

[13] O. Dahary, O. Patashnik, K. Aberman, and D. Cohen-Or. Be Yourself: Bounded Attention for Multi-subject Text-to-Image Generation. In *Proceedings of the European Conference on Computer Vision*, volume 15072 of *Lecture Notes in Computer Science*, pages 432–448. Springer, 2024.

[14] dewei Zhou, J. Xie, Z. Yang, and Y. Yang. 3DIS: Depth-Driven Decoupled Image Synthesis for Universal Multi-Instance Generation. In *Proceedings of the International Conference on Learning Representations*, 2025. URL `https://openreview.net/forum?id=MagmwodCAB`.

[15] P. Esser, S. Kulal, A. Blattmann, R. Entezari, J. Müller, H. Saini, Y. Levi, D. Lorenz, A. Sauer, F. Boesel, D. Podell, T. Dockhorn, Z. English, and R. Rombach. Scaling Rectified Flow Transformers for High-Resolution Image Synthesis. In *Proceedings of the International Conference on Machine Learning*. OpenReview.net, 2024.

[16] Y. Feng, B. Gong, D. Chen, Y. Shen, Y. Liu, and J. Zhou. Ranni: Taming Text-to-Image Diffusion for Accurate Instruction Following. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4744–4753. IEEE, 2024.

[17] P. Gao, L. Zhuo, D. Liu, R. Du, X. Luo, L. Qiu, Y. Zhang, C. Lin, R. Huang, S. Geng, R. Zhang, J. Xi, W. Shao, Z. Jiang, T. Yang, W. Ye, H. Tong, J. He, Y. Qiao, and H. Li. Lumina-T2X: Transforming Text into Any Modality, Resolution, and Duration via Flow-based Large Diffusion Transformers, 2024. URL https://arxiv.org/abs/2405.05945.

[18] B. Gong, S. Huang, Y. Feng, S. Zhang, Y. Li, and Y. Liu. Check, Locate, Rectify: A Training-Free Layout Calibration System for Text- to- Image Generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6624–6634. IEEE, 2024.

[19] R. He, B. Cheng, Y. Ma, Q. Jia, S. Liu, A. Ma, X. Wu, L. Wu, D. Leng, and Y. Yin. PlanGen: Towards Unified Layout Planning and Image Generation in Auto-Regressive Vision Language Models, 2025. URL https://arxiv.org/abs/2503.10127.

[20] Y. He, R. Salakhutdinov, and J. Z. Kolter. Localized text-to-image generation for free via cross attention control, 2023. URL https://arxiv.org/abs/2306.14636.

[21] J. Hessel, A. Holtzman, M. Forbes, R. L. Bras, and Y. Choi. CLIPScore: A Reference-free Evaluation Metric for Image Captioning. In *Proceedings of the Empirical Methods in Natural Language Processing*, pages 7514–7528. Association for Computational Linguistics, 2021.

[22] J. Ho, A. Jain, and P. Abbeel. Denoising Diffusion Probabilistic Models. In *NeurIPS*, 2020.

[23] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. LoRA: Low-Rank Adaptation of Large Language Models. In *Proceedings of the International Conference on Learning Representations*. OpenReview.net, 2022.

[24] X. Huang, Y.-J. Huang, Y. Zhang, W. Tian, R. Feng, Y. Zhang, Y. Xie, Y. Li, and L. Zhang. Open-Set Image Tagging with Multi-Grained Text Supervision, 2023. URL https://arxiv.org/abs/2310.15200.

[25] C. Jia, M. Luo, Z. Dang, G. Dai, X. Chang, M. Wang, and J. Wang. SSMG: Spatial-Semantic Map Guided Diffusion Model for Free-Form Layout-to-Image Generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 2480–2488. AAAI Press, 2024.

[26] Y. Kirstain, A. Polyak, U. Singer, S. Matiana, J. Penna, and O. Levy. Pick-a-Pic: An Open Dataset of User Preferences for Text-to-Image Generation. In *Advances in Neural Information Processing Systems*, 2023.

[27] Y. Lee, T. Yoon, and M. Sung. GrounDiT: Grounding Diffusion Transformers via Noisy Patch Transplantation. In *Advances in Neural Information Processing Systems*, 2024.

[28] Y. Li, H. Liu, Q. Wu, F. Mu, J. Yang, J. Gao, C. Li, and Y. J. Lee. GLIGEN: Open-Set Grounded Text-to-Image Generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22511–22521. IEEE, 2023.

[29] Y. Li, M. Keuper, D. Zhang, and A. Khoreva. Adversarial Supervision Makes Layout-to-Image Diffusion Models Thrive. In *Proceedings of the International Conference on Learning Representations*. OpenReview.net, 2024.

[30] Z. Li, J. Zhang, Q. Lin, J. Xiong, Y. Long, X. Deng, Y. Zhang, X. Liu, M. Huang, Z. Xiao, D. Chen, J. He, J. Li, W. Li, C. Zhang, R. Quan, J. Lu, J. Huang, X. Yuan, X. Zheng, Y. Li, J. Zhang, C. Zhang, M. Chen, J. Liu, Z. Fang, W. Wang, J. Xue, Y. Tao, J. Zhu, K. Liu, S. Lin, Y. Sun, Y. Li, D. Wang, M. Chen, Z. Hu, X. Xiao, Y. Chen, Y. Liu, W. Liu, D. Wang, Y. Yang, J. Jiang, and Q. Lu. Hunyuan-DiT: A Powerful Multi-Resolution Diffusion Transformer with Fine-Grained Chinese Understanding, 2024. URL https://arxiv.org/abs/2405.08748.

[31] T. Lin, M. Maire, S. J. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: Common Objects in Context. In *Proceedings of the European Conference on Computer Vision*, volume 8693 of *Lecture Notes in Computer Science*, pages 740–755. Springer, 2014.

[32] Z. Lin, D. Pathak, B. Li, J. Li, X. Xia, G. Neubig, P. Zhang, and D. Ramanan. Evaluating Text-to-Visual Generation with Image-to-Text Generation. In *Proceedings of the European Conference on Computer Vision*, volume 15067 of *Lecture Notes in Computer Science*, pages 366–384. Springer, 2024.

[33] Y. Lipman, R. T. Q. Chen, H. Ben-Hamu, M. Nickel, and M. Le. Flow matching for generative modeling. In *Proceedings of the International Conference on Learning Representations*. OpenReview.net, 2023.

[34] B. Liu, E. Akhgari, A. Visheratin, A. Kamko, L. Xu, S. Shrirao, C. Lambert, J. Souza, S. Doshi, and D. Li. Playground v3: Improving Text-to-Image Alignment with Deep-Fusion Large Language Models, 2024. URL `https://arxiv.org/abs/2409.10695`.

[35] S. Liu, Z. Zeng, T. Ren, F. Li, H. Zhang, J. Yang, Q. Jiang, C. Li, J. Yang, H. Su, J. Zhu, and L. Zhang. Grounding DINO: Marrying DINO with Grounded Pre-training for Open-Set Object Detection. In *Proceedings of the European Conference on Computer Vision*, volume 15105 of *Lecture Notes in Computer Science*, pages 38–55. Springer, 2024.

[36] Z. Lv, Y. Wei, W. Zuo, and K. K. Wong. PLACE: Adaptive Layout-Semantic Fusion for Semantic Image Synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9264–9274. IEEE, 2024.

[37] M. Ohanyan, H. Manukyan, Z. Wang, S. Navasardyan, and H. Shi. Zero-Painter: Training-Free Layout Control for Text-to-Image Synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8764–8774. IEEE, 2024.

[38] openfree. flux-chatgpt-ghibli-lora. `https://huggingface.co/openfree/flux-chatgpt-ghibli-lora`, 2025.

[39] PatrickStarrrr. FLUX - Oil painting. `https://civitai.com/models/1455014/chatgpt-4o-renderer?modelVersionId=1697982`, 2024.

[40] W. Peebles and S. Xie. Scalable Diffusion Models with Transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4172–4182. IEEE, 2023.

[41] Q. Phung, S. Ge, and J. Huang. Grounded Text-to-Image Synthesis with Attention Refocusing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7932–7942. IEEE, 2024.

[42] D. Podell, Z. English, K. Lacey, A. Blattmann, T. Dockhorn, J. Müller, J. Penna, and R. Rombach. SDXL: Improving Latent Diffusion Models for High-Resolution Image Synthesis. In *Proceedings of the International Conference on Learning Representations*. OpenReview.net, 2024.

[43] Qwen Team. Qwen2.5-VL, January 2025. URL `https://qwenlm.github.io/blog/qwen2.5-vl/`.

[44] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen. Hierarchical Text-Conditional Image Generation with CLIP Latents, 2022. URL `https://arxiv.org/abs/2204.06125`.

[45] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-Resolution Image Synthesis with Latent Diffusion Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10674–10685. IEEE, 2022.

[46] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Medical Image Computing and Computer-Assisted Intervention*, volume 9351 of *Lecture Notes in Computer Science*, pages 234–241. Springer, 2015.

[47] C. Saharia, W. Chan, S. Saxena, L. Li, J. Whang, E. L. Denton, S. K. S. Ghasemipour, R. G. Lopes, B. K. Ayan, T. Salimans, J. Ho, D. J. Fleet, and M. Norouzi. Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding. In *Advances in Neural Information Processing Systems*, 2022.

[48] T. Shirakawa and S. Uchida. NoiseCollage: A Layout-Aware Text-to-Image Diffusion Model Based on Noise Cropping and Merging. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8921–8930. IEEE, 2024.

[49] stability.ai. Stable Diffusion 3.5. `https://stability.ai/news/introducing-stable-diffusion-3-5`, Nov 2024.

[50] A. Taghipour, M. Ghahremani, M. Bennamoun, A. M. Rekavandi, H. Laga, and F. Boussaid. Box It to Bind It: Unified Layout Control and Attribute Binding in T2I Diffusion Models, 2024. URL `https://arxiv.org/abs/2402.17910`.

[51] M. Tancik, P. P. Srinivasan, B. Mildenhall, S. Fridovich-Keil, N. Raghavan, U. Singhal, R. Ramamoorthi, J. T. Barron, and R. Ng. Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains. In *Advances in Neural Information Processing Systems*, 2020.

[52] vjleoliu. ChatGPT-4o Renderer. `https://civitai.com/models/1455014/chatgpt-4o-renderer?modelVersionId=1697982`, 2025.

[53] X. Wang, T. Darrell, S. S. Rambhatla, R. Girdhar, and I. Misra. InstanceDiffusion: Instance-Level Control for Image Generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6232–6242. IEEE, 2024.

[54] Y. Wu, X. Zhou, B. Ma, X. Su, K. Ma, and X. Wang. IFAdapter: Instance Feature Control for Grounded Text-to-Image Generation, 2024. URL `https://arxiv.org/abs/2409.08240`.

[55] J. Xie, Y. Li, Y. Huang, H. Liu, W. Zhang, Y. Zheng, and M. Z. Shou. BoxDiff: Text-to-Image Synthesis with Training-Free Box-Constrained Diffusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7418–7427. IEEE, 2023.

[56] J. Xu, X. Sun, Z. Zhang, G. Zhao, and J. Lin. Understanding and Improving Layer Normalization. In *Advances in Neural Information Processing Systems*, pages 4383–4393, 2019.

[57] H. Xue, Z. Huang, Q. Sun, L. Song, and W. Zhang. Freestyle Layout-to-Image Synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14256–14266. IEEE, 2023.

[58] B. Yang, Y. Luo, Z. Chen, G. Wang, X. Liang, and L. Lin. LAW-Diffusion: Complex Scene Generation by Diffusion with Layouts. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22612–22622. IEEE, 2023.

[59] Z. Yang, J. Wang, Z. Gan, L. Li, K. Lin, C. Wu, N. Duan, Z. Liu, C. Liu, M. Zeng, and L. Wang. ReCo: Region-Controlled Text-to-Image Generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14246–14255. IEEE, 2023.

[60] Y. Yao, T. Yu, A. Zhang, C. Wang, J. Cui, H. Zhu, T. Cai, H. Li, W. Zhao, Z. He, Q. Chen, H. Zhou, Z. Zou, H. Zhang, S. Hu, Z. Zheng, J. Zhou, J. Cai, X. Han, G. Zeng, D. Li, Z. Liu, and M. Sun. MiniCPM-V: A GPT-4V Level MLLM on Your Phone, 2024. URL `https://arxiv.org/abs/2408.01800`.

[61] H. Zhang, D. Hong, Y. Wang, J. Shao, X. Wu, Z. Wu, and Y.-G. Jiang. CreatiLayout: Siamese Multimodal Diffusion Transformer for Creative Layout-to-Image Generation, 2025. URL `https://arxiv.org/abs/2412.03859`.

[62] L. Zhang, A. Rao, and M. Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3813–3824. IEEE, 2023.

[63] G. Zheng, X. Zhou, X. Li, Z. Qi, Y. Shan, and X. Li. LayoutDiffusion: Controllable Diffusion Model for Layout-to-Image Generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22490–22499. IEEE, 2023.

[64] D. Zhou, Y. Li, F. Ma, X. Zhang, and Y. Yang. MIGC: Multi-Instance Generation Controller for Text-to-Image Synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6818–6828. IEEE, 2024.

[65] D. Zhou, Y. Li, F. Ma, Z. Yang, and Y. Yang. MIGC++: Advanced Multi-Instance Generation Controller for Image Synthesis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(3):1714–1728, 2025.

[66] D. Zhou, J. Xie, Z. Yang, and Y. Yang. 3DIS-FLUX: simple and efficient multi-instance generation with DiT rendering, 2025. URL `https://arxiv.org/abs/2501.05131`.

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: We introduce our InstanceAssemble method in the abstract and introduction, provide a detailed description in Section 3, and present experimental results in Section 4.

   Guidelines:

   - The answer NA 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 NA 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 the limitations of this work in Section 5.

   Guidelines:

   - The answer NA 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: [NA]

Justification: This paper does not include theoretical results.

Guidelines:

- The answer NA 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 architecture of our module is introduced in Section 3, and the implementation details are presented at the beginning of Section 4.

   Guidelines:

   - The answer NA 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: We will release our code upon acceptance.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.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://nips.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, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: The implementation details are presented at the beginning of Section 4.

   Guidelines:

   - The answer NA 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: Although we report multiple evaluation metrics (e.g., mIoU, VQA, CLIP) showing significant improvements over baselines, we did not include error bars, standard deviations, confidence intervals, or statistical significance tests across multiple runs because it would be too computationally expensive. As such, our results do not fully meet the criteria outlined in this question for statistical rigor and uncertainty quantification.

   Guidelines:

   - The answer NA 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: We present the computer resources needed for the experiments at the begining of Section 4.

   Guidelines:

   - The answer NA 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: We conducted the research in the paper conform, in every respect, with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA 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: We discussed the broader impacts in the last part of the article.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA 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., pretrained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: We use the existing excellent SD3, Flux.1 and their safeguards. Our model will not bring more risk than SD3 or Flux.1.

    Guidelines:

    - The answer NA 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: We work with open-source models that are publicly available, and we cited them properly.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

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

13. **New assets**

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

    Answer: [Yes]

    Justification: The dataset defined for our benchmark will be made publicly available, in case of acceptance, together with the documentation required for reproducing the experiments. Moreover, in case of acceptance, we will also release the source code of our model.

    Guidelines:

    - The answer NA 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: [NA]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA 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: [NA]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA 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 rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer NA 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 (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

Supplementary Material

A Early-Stage Layout Control in Assemble-MMDiT

We apply the Assemble-MMDiT layout control module exclusively during the initial 30% of the denoising trajectory and deactivate it for the remaining 70%. This two-stage strategy provides robust low-frequency structural guidance in the early stages to facilitate layout alignment, while allowing subsequent unconstrained refinement of high-frequency details during later denoising phases.

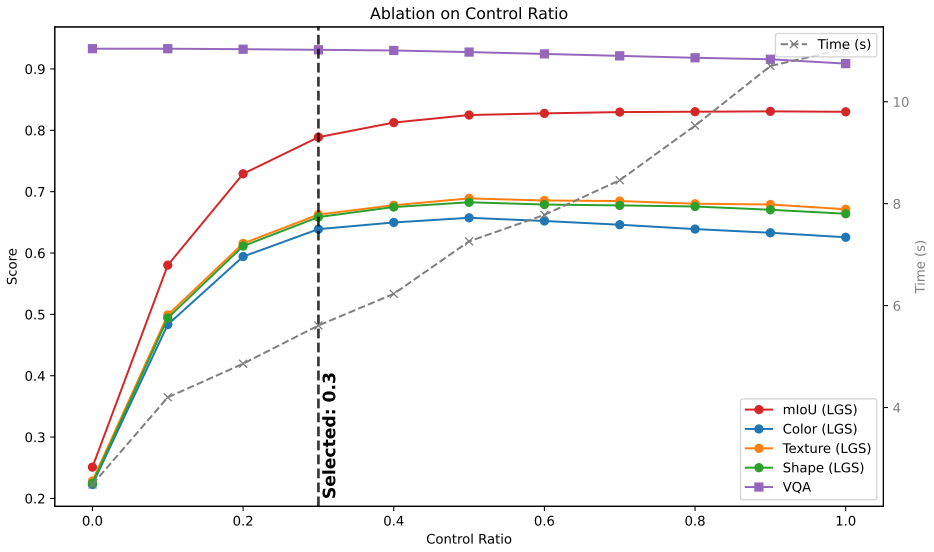


Figure 9: Impact of the proportion of diffusion steps incorporating layout conditioning on generation quality.

As illustrated in Figure 9, restricting layout control to less than 30% of the diffusion process results in insufficient layout alignment with the target bounding boxes. In contrast, extending control beyond this optimal threshold leads to a decline in output quality. Furthermore, increasing the proportion of layout-guided steps results in significant additional computational cost.

B Additional Ablation Studies

B.1 Effect of Bbox Encoding and DenseSample

To clarify the role of bounding box encoding and DenseSample, we further ablated the SD3-M based InstanceAssemble model on DenseLayout. Bounding box embeddings guide correct object placement, while DenseSample provides additional improvements in spatial accuracy and instance-level semantics. The results in Table 6 demonstrate that both components contribute to the overall performance.

Table 6: Ablation on bounding box encoding and DenseSample.

Setting	mIoU↑	color↑	texture↑	shape↑	VQA↑
w/o bbox encoding, w/o DenseSample	51.22	32.15	34.04	33.53	93.30
w/ bbox encoding, w/o DenseSample	51.28	32.68	34.94	34.58	93.33
w/ bbox encoding, w/ DenseSample	**52.07**	**33.77**	**36.21**	**35.81**	**93.54**

B.2 Comparison with Attention Mask-based Region Injection

We also compare our Assemble-Attn design with attention mask-based region injection. While both can be viewed as region-wise attention mechanisms, attention masks operate globally and may cause

semantic leakage in overlapping regions. Our method instead applies instance-wise self-attention on cropped latent regions and then fuses the updated features via the Assemble step, which is more effective in dense layouts. As shown in Table 7, our design achieves superior instance attribute consistency and a higher VQA score compared to the attention mask baseline.

Table 7: Comparison between attention mask-based injection and our Assemble-Attn.

Method	spatial↑	color↑	texture↑	shape↑	VQA↑
SD3-Medium (base model)	77.49	60.28	62.55	60.38	93.30
Attention mask (SD3-M)	94.11	74.28	77.58	76.54	91.53
InstanceAssemble (ours, SD3-M)	**94.97**	**77.53**	**80.72**	**80.11**	**93.12**

C Underlying data for radar-chart visualizations

In Sections 4.3, we utilized a radar chart to depict each quantitative variable along equi-angular axes, providing an intuitive comparison. This visualization highlights the multifaceted superiority of our method. Here, we present the corresponding raw evaluation results in tabular form. Specifically, Tab. 8 corresponds to Fig. 6, thus ensuring a clear mapping between each radar-chart subfigure and its underlying data.

Table 8: **Quantitative results of additional visual content on DenseLayout.**

DenseLayout	Layout Grounding Score				Global Quality		
	mIoU↑	color↑	texture↑	shape↑	VQA↑	Pick↑	CLIP↑
Real Images(Upper Bound)	92.35	76.52	80.78	79.78			
text	43.72	26.57	28.56	28.39	**93.37**	21.63	12.45
text+image	**55.29**	**42.15**	**44.50**	**44.24**	91.66	**22.05**	12.95
text+depth	49.64	28.25	31.82	31.62	92.83	21.28	13.25
text+edge	50.73	29.45	33.92	33.84	90.13	21.26	**13.55**

D More Details on DenseLayout Evaluation Dataset

D.1 Construction Pipeline of DenseLayout Dataset

The DenseLayout dataset is constructed through a multi-stage pipeline designed to extract high-density and semantically-rich layout information from synthetic images. The pipeline includes following steps:

1. **Image Generation using Flux.1-Dev [4]**
   A diverse set of synthetic images is generated using Flux.1-Dev, a text-to-image model. The input prompts are generic textual descriptions, sampled from the LayoutSAM dataset, which is based on SA-1B. The images are resized to maintain the same aspect ratio as the original SA-1B images, with the longer edge set to 1024 pixels. This step provides a visually complex base for extracting layout structures.
2. **Multi-label Tagging using RAM++ [24]**
   The generated images are tagged using RAM++, the next-generation model of RAM, which supports open-set recognition. These tags offer high-level semantic guidance for subsequent grounding.
3. **Object Detection via GroundingDINO [35]**
   Using the image and its predicted tags as input, GroundingDINO performs open-set object detection. It outputs bounding boxes and class labels for all detected entities. The detection is configured with a `box_threshold` of 0.35 and a `text_threshold` of 0.25. Each detected bounding box is treated as an **instance bounding box**, and the corresponding predicted label is recorded as the **instance description**.
4. **Detailed Captioning with Qwen2.5-VL [43]**
   Each bounding box region is cropped from the original image and fed into Qwen2.5-VL

to generate a fine-grained caption. These region-level captions are stored as the **detailed description** for each instance, enriching the semantic information beyond category labels.

5. **Density Filtering**
   To ensure high layout complexity, only images with 15 or more detected instances (as output by GroundingDINO) are retained. This results in a dense layout distribution suitable for layout-conditioned generation tasks. The distribution of instance count is shown in Fig. 10.

Finally, the DenseLayout dataset contains **5,000 images** and **90,339 instances**, with an average of **18.1 instances per image**.

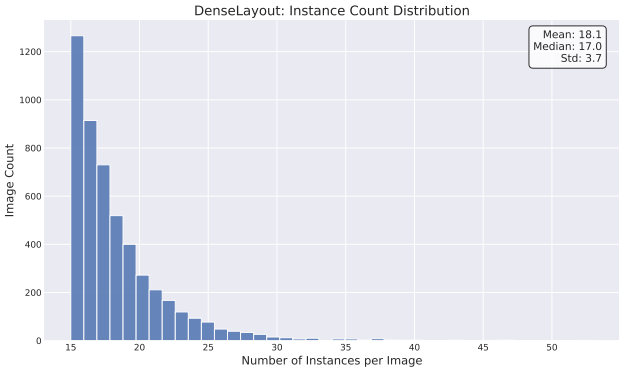
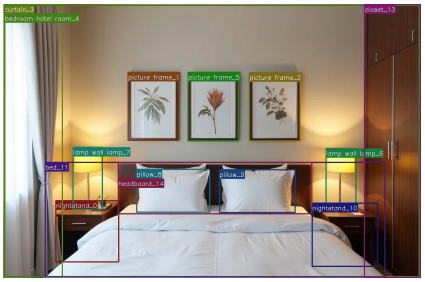


Figure 10: Instance count distribution per image in DenseLayout.

Annotation Format. The annotation for each image consists of:

- `global_caption`: the original prompt used for image generation.
- `image_info`: metadata of the image, including height and width.
- `instance_info`: a list of instances, each with:
  - `bbox`: the bounding box of the instance, formatting as $[x_1, y_1, x_2, y_2]$.
  - `description`: the category label predicted by GroundingDINO.
  - `detail_description`: a fine-grained caption generated by Qwen2.5-VL for the cropped region.

D.2 Samples of DenseLayout Dataset



```
"instance_info": [
{
    "bbox": [129,489,283,642],
    "description": "nightstand",
    "detail_description": "The nightstand is dark brown,
         compact, with a drawer."
},
{
    "bbox": [306,170,430,339],
    "description": "picture frame",
    "detail_description": "Brown wooden frame containing a
         watercolor painting of green leaves on a white
         background."
},
{
    "bbox": [603,170,727,340],
    "description": "picture frame",
    "detail_description": "A simple brown wooden frame holds
          a botanical print with detailed leaves and stems."
},
...
]
```

Figure 11: A sample of DenseLayout and its annotation.

Figure 12: Representative samples from DenseLayout dataset demonstrating: (a) High-density scene with ≥ 15 instances, (b) Complex instance relationships with precise attribute specifications.

E More results with textual-only content

E.1 InstanceAssemble based on SD3-Medium

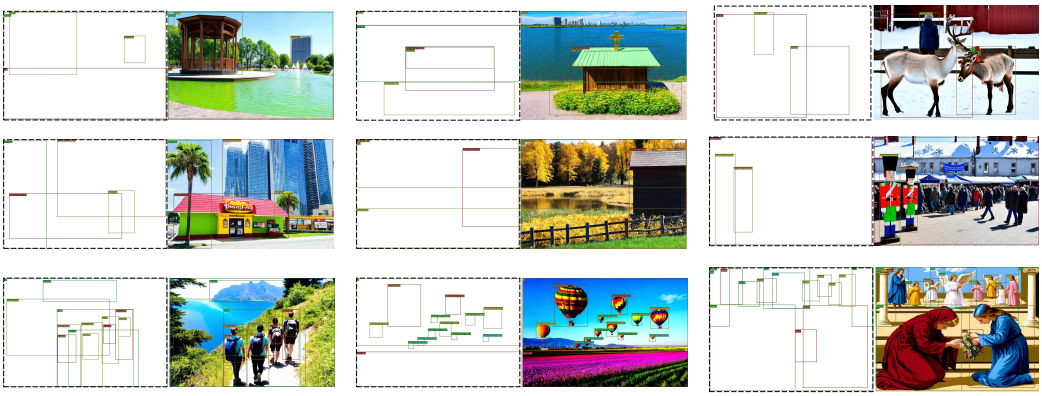


Figure 13: More results of InstanceAssemble based on SD3-Medium with textual-only content.

E.2 InstanceAssemble based on Flux.1-Dev

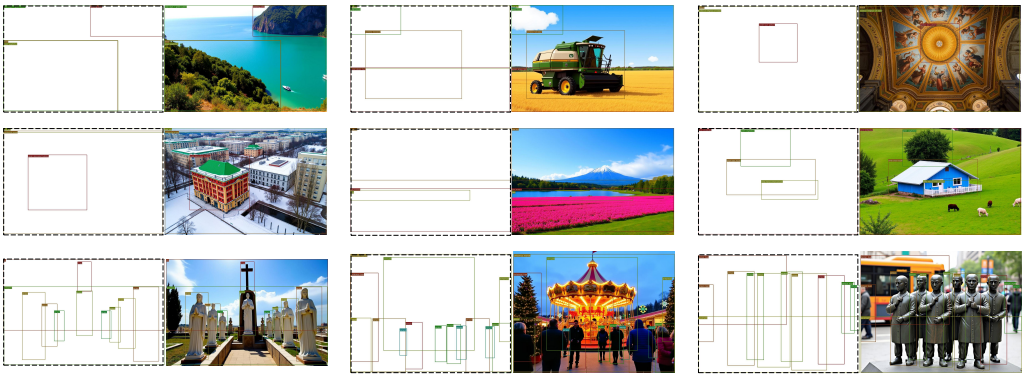


Figure 14: More results of InstanceAssemble based on Flux.1-Dev with textual-only content.

E.3 InstanceAssemble based on Flux.1-Schnell

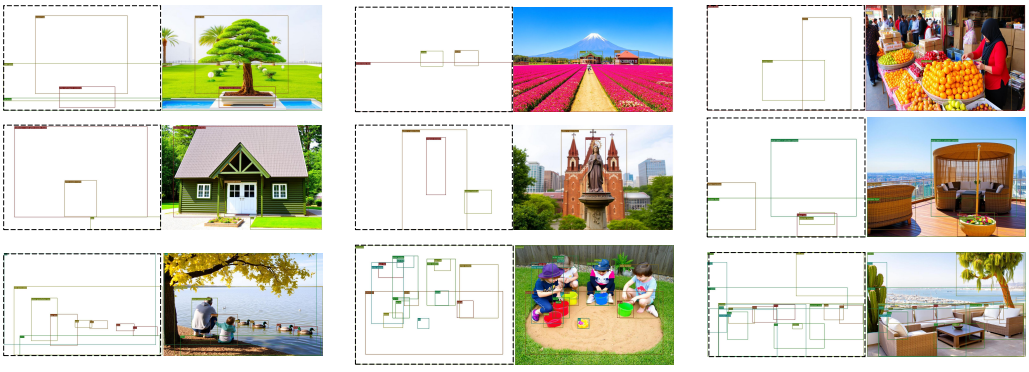


Figure 15: More results of InstanceAssemble based on Flux.1-Schnell with textual-only content.

F More results with additional visual content

F.1 Additional Image

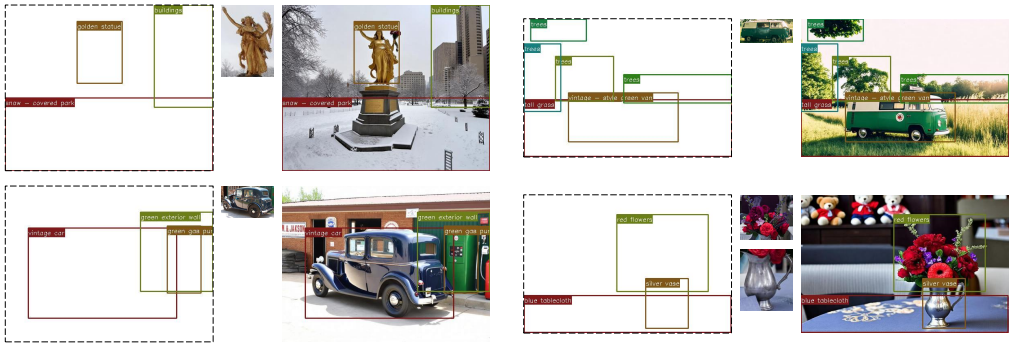


Figure 16: More results with additional image.

F.2 Additional Depth

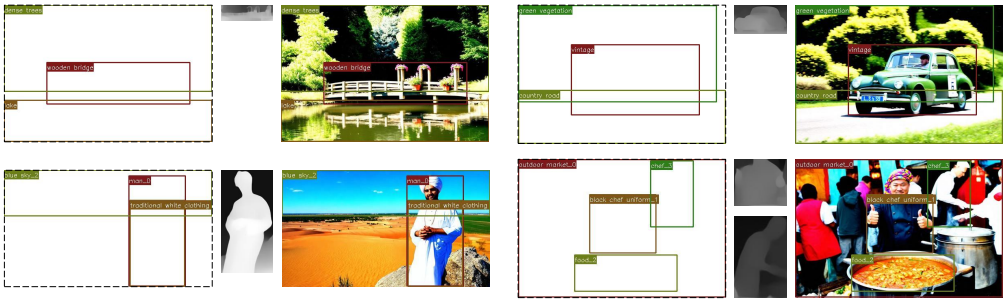


Figure 17: More results with additional depth.

F.3 Additional Edge

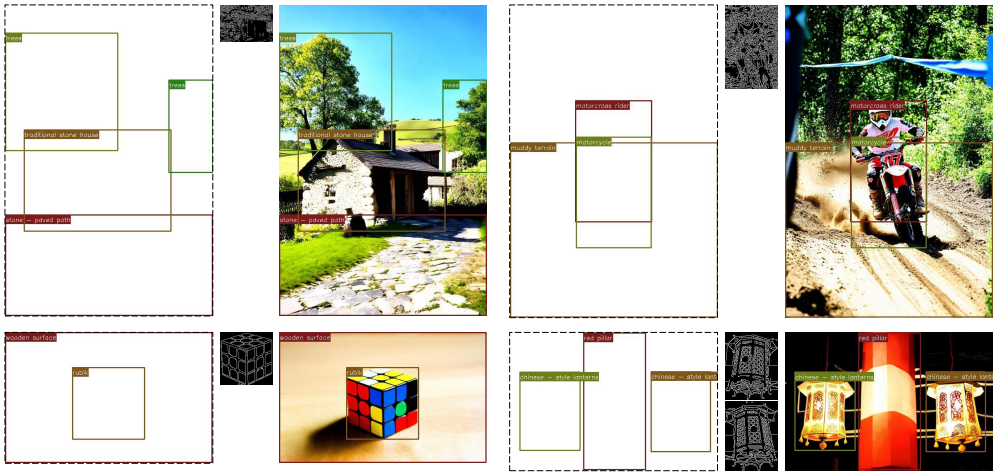


Figure 18: More results with additional edge.