Uni-SD: Unified Image Generation for Object Detection

Xinyu Zhang[1] **Xiaodi Wang**[1] **Dong Gong**[2] **Jinwen Chen**[1] **Haixiao Yue**[1] **Gang Zhang**[1]
Kun Yao[1] **Lingqiao Liu**[3] **Errui Ding**[1] **Jingdong Wang**[1]
[1]Baidu VIS [2]University of New South Wales [3]The University of Adelaide
{zhangxinyu14,wangjingdong}@baidu.com

Abstract

Text-to-image generation has shown great success on synthesis visually plausible images. Given descriptions of the semantic contents and previous layout, the synthetic images generated from diffusion models also show great potential on improving object detection. To extend the application into more realistic scenario, it is necessary to produce more precise control of objects and enrich the diversity of the generated objects. Precise control of the generation and diversity enhancement usually need to produced via various types of instruction inputs. In this paper, we present Uni-SD, an image generation pipeline to jointly consider these two requirements in a unified model. Uni-SD accepts both layout and instruction inputs in a unified manner, which is beneficial for the flexibility of use and keeps the generation's precision and diversity. To enhance the generation ability and response on these two input formats, we design a task-aware rectified cross-attention module and learn task-aware gated embeddings to decouple these two formats during the generation. With synthetic data from Uni-SD, we show consistent improvements on the object detection task. On LVIS val, Uni-SD improves 1.17% and 4.89% Box AP and rare AP, and 1.17% and 5.68% Mask AP and rare AP over the baseline, showing a great potential for the object detection and instance segmentation task, especially on the long-tail setting.

1 Introduction

Diffusion models [21, 43, 38, 40, 37, 36, 4, 34, 25] represent a pivotal development in generative technology, offering superior capabilities in generating high-fidelity, realistic images. Some works [20, 49, 32, 2, 11, 5, 28, 14, 55, 57, 42] have explored to utilize synthetic images from diffusion models to capture the intricate distributions of realistic data, presenting new opportunities for synthetic data utilization in the real-world scenario.

Recently, researchers and developers attempt to study how effectiveness of generated images on perception tasks [27, 23, 3, 19, 41, 48, 58, 46, 50], *e.g.*, image recognition and object detection. We target on the object detection task in this paper, and naturally ask two questions: *i) whether the synthetic images are useful or not to improve the performance? ii) if yes, what kind of synthetic images are useful?*

Current methods [54, 28, 10, 58, 46, 8, 55] applying synthetic images on object detection can be classified into two categories, *i.e.*, layout-based methods and instruction-based methods. The former approaches [54, 28, 10, 58, 46, 53] input pre-defined layouts with masks or bounding boxes as the guidance for image composition. It is relatively accurate to generate expected objects, while resulting in extra complexity due to the manual layout design. Besides this, layout-based input is hard to scale up and the limited layout prevents the generation diversity. Although [47, 8, 55, 9] can automatically obtain layouts, it is still difficult to guarantee of reliability, including objects' size, shape, position

Preprint. Under review.

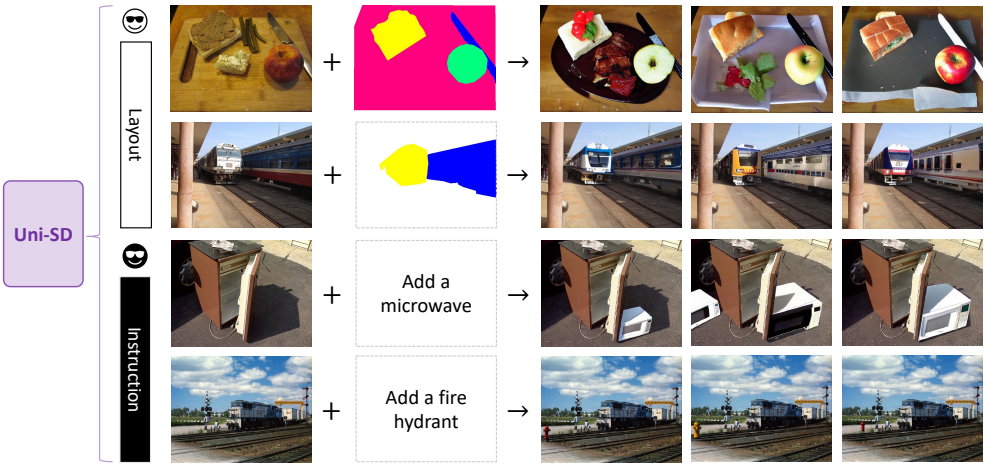


Figure 1: Our Uni-SD is a unified image generation pipeline, supporting both layout-formatted and instruction-formatted inputs. The last three synthetic images are sampled from different seed. (Zoom in for best view)

and relationships. The latter instruction-based methods [14, 50, 5, 57] alternatively change to use user-friendly instruction for generation. These methods are able to edit images flexibly without costly layout design, however, they inevitably change the image's original information and fail to accurately execute user's instructions.

In this paper, we propose Uni-SD, a novel unified image generation pipeline, taking the essence of layout-based and instruction-based methods. Uni-SD supports both layout- and instruction-formatted inputs, and outputs generated images according to the input format. Uni-SD enables precise control over the generated content and is flexible for the user use. Specially, we present an advanced module, which is inherited from [53], named task-aware rectified cross-attention (TRCA). TRCA can not only enhance each text token on specified region when using layout input format, but also keep understanding the whole text prompt for the instruction input. To further improve the learning ability of each input format, we introduce task-aware gated embeddings (TGE) to guide the generation process. TGE are learnable parameters, which are updated during the training process. During the inference, we choose which embedding to use to perform either layout or instruction inputs.

To verify whether synthetic images from Uni-SD are useful or not on object detection, we perform experimental evaluation qualitatively and quantitatively. Following [16, 58, 46], we test the effectiveness of generated images when applying them to train detectors. Experiments show that Uni-SD brings consistent improvements on COCO and LVIS. Meanwhile, to explore what kind of synthetic images are beneficial, we generate frequent, common and rare classes in LVIS, respectively. We find that Uni-SD achieves larger improvement when generating more rare-class images, showing that Uni-SD is beneficial for the long-tail setting task. In summary, our contributions are:

- Introduce a unified image generation pipeline Uni-SD, supporting both layout and instruction inputs to enable precise control over the generated content and is flexible for the user use.
- Propose a task-aware rectified cross-attention module and task-aware gated embeddings to enhance the attention operation and learning ability of each input format, simultaneously.
- Verify the effectiveness of Uni-SD in both qualitative and quantitative experiments, and show that synthetic images from Uni-SD can bring improvements on object detection, especially on the long-tail classes.

2 Related Work

Image generation with diffusion models. have garnered significant interest following the development of large pretrained models such as Stable Diffusion (SD) [38], Imagen[40], DALL-E [37, 36, 4], and SDXL [34]. These models are increasingly being applied to specific uses, such as personalized image generation [24, 15, 39, 52] and local image editing [20, 5, 49, 2, 56, 54, 10]. These methods can be classified into two categories, *i.e.*, layout-formatted [20, 49, 2, 56, 28, 54, 31, 10, 55, 53] and instruction-formatted methods [5, 57, 42, 14, 44, 6, 20].

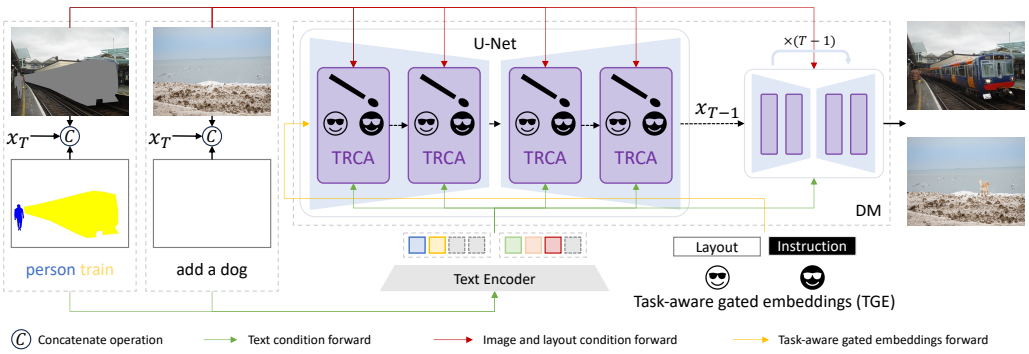


Figure 2: The overview of our Uni-SD pipeline, supporting both layout- and instruction-formatted inputs. Uni-SD has two main modules, *i.e.*, task-aware rectified cross-attention (TRCA) and task-aware gated embeddings (TGE). TRCA can control the cross-attention map operation according to the input format, and TGE are learnable embeddings to guide the image generation toward a correct direction for different types of inputs. The text encoder and the diffusion model (DM) are from the stable diffusion model. For simplicity, we omit the autoencoder in the figure.

For layout-formatted methods, researchers have discovered to introduce masks into the diffusion models. Methods [20, 49] focus on modifying attention maps from spatial features and the text prompt for image editing. However, the implicit masks from attention maps limits the ability to generate completely independent targets, tending instead to modify ancillary elements like clothing on animals. Instead, other methods [2, 56, 28, 54, 31, 10, 55] alter to use prior masks to guide the generation, showing more accurate generation. The underlying drawback of layout-formatted methods is the time-cost mask priors, limiting to generate tremendous images for other tasks.

For instruction-formatted methods, some methods [5, 57, 42, 14, 44] constructs a serious annotated dataset with automatic ways [6, 20] or human annotation for instruction-guided image editing. These methods are easy for user use, while usually suffering from inconsistency problem, *i.e.*, in-context information in the original images will be also modified, which does not satisfy user's requirements.

Synthetic data for perception tasks. Recent studies have been developed to integrate synthetic data into natural perception tasks [3, 19, 41, 13, 12, 16, 46], *e.g.*, image classification and object detection. Here, we focus on object detection task.

Some layout-formatted methods, like [13, 12, 16, 58, 46], input pre-defined layouts with masks or bounding boxes as the guidance for image composition. These methods largely rely on layout design, which is time consumption. Although [47, 8, 55, 9] can automatically obtain layouts, challenges persist with the integration of synthetic data, particularly in how pasted objects are retained, and how is the quality of objects' size, shape, position and relationships. The unrealistic generated images can inversely hinder the implicit modeling of context, ultimately diminishing detection accuracy [13, 12]. To address these issues, [50] try to direct obtain synthetic data from instructions. However, they only focus on adding a object, which has limited diversity.

Our proposed Uni-SD aims to combine the advantages of both layout-formatted and instruction-formatted methods into a unified model, and generate useful synthetic data for object detection.

3 Uni-SD

Our work focuses on improving the performance of object detection using generated images. We try to explore i) whether synthetic images are really useful or not to improve the performance and ii) what kind of synthetic images are truly useful.

3.1 Preliminary: great potential to task the essence

Existing methods process image generation through structural design like attention maps [20, 39, 49, 24], human reference priors like layout with bounding boxes [32, 54, 28, 10, 55, 46], and instruction-based tuning with data pairs [5, 57, 42, 50, 14, 45]. They have their own advantages and

disadvantages. Layout-based methods can accurately generate expected objects, while the manual layout design is time consuming and display limited diversity. Instruction-based methods is more flexible for the user use, while the consistency can not be guaranteed. In other works, these methods will inevitably change the areas in the images that we do not want to modify.

Some layout-based methods [16, 58, 46] has attempted to use synthetic data for object detection. They clearly show that synthetic data can truly improve the performance when adding these generated data into the detection training. However, they still ignore the analyses of how to use these synthetic data more reasonably, and the layout diversity and rationality can not be well guaranteed. We also try to generate images using instruction-based method [5], the performance drops a lot (see Table 2).

This motivates us to consider to generate useful synthetic data in a controllable way for object detection. Recall the advantages of layout-based and instruction-based methods in the above, we consider to design a unified image generation pipeline to task the essence of both methods.

3.2 Uni-SD

To make the goal success, we present Uni-SD, a novel unified image generation pipeline. Uni-SD supports both layout and instruction input formats, which meets expectation of absorbing advantages of layout-formatted and instruction-formatted methods. Uni-SD can not only accurately control what we want using layout or instruction, but also keep consistency of original images for more realistic. That is to say, we can generate synthetic data what we want. In Uni-SD, we design a task-aware rectified cross-attention (TRCA) and task-aware gated embeddings (TGE) to better execute these two separate inputs. The following section will introduce the details one-by-one, and the overview pipeline of Uni-SD is shown in Figure 2.

3.2.1 Task-aware rectified cross-attention

Previous work [53] designs a rectified cross-attention (RCA) to enforce each text token to act on the pixels in a specified region. Given a specific layout, RCA can generate realistic and creative images with a wide range of semantics provided by text prompts. Inspired by [53], we advance a task-aware rectified cross-attention (TRCA), an upgraded version of RCA that can not only act on specific-region pixels of attention maps, but also reserve the original cross-attention ability to follow the whole text prompt. Specifically, we unify the layout- and instruction-formatted inputs in a unified diffusion model, including three key factors: representing semantics, task-aware conditions and task-aware rectified cross-attention operation.

Representing semantics. For the layout input, we follow [53] to describe each semantic (*e.g.*, an object class from COCO [29]) with a word (*e.g.*, "cat") or a phrase (*e.g.*, "teddy bear") as a concept. The concepts in each image are stacked into a text prompt (*e.g.*, "cat teddy bear apple") as the input of the text encoder. For the instruction input, we simply use original text prompt without any modification as the input. Note that as we focus on improving object detection, it is required to add more needed objects into the real-world images. Therefore, the text prompt is formatted into "add <num> <objs>", where "<objs>" represents the name of the inserted objects and "<num>" is the quantity. These two types of text embeddings from the text encoder serve as the semantics to perform the following generation.

Task-aware conditions. For the layout input, the condition is the given layout, *e.g.*, masks or bounding boxes of concepts. For the instruction input, there is no additional condition; therefore, we use a binary map with all values as 1 as the condition.

Task-aware rectified cross-attention operation. The original cross-attention layer tasks two inputs, including text embeddings and image features. The text embeddings act as keys and values and the image features are as queries to calculate cross-attention maps. The channel of attention maps can somewhat reflect the specific text prompt [22, 20]. RCA [53] thus rectify the attention map of each text embedding using the corresponding mask to determine the spatial layout of the generated image. For an individual text concept, its semantic mask $S \in \mathbb{R}^{H \times W}$ is used as the binary map to modify the associated attention map $\mathcal{M} \in \mathbb{R}^{H \times W}$ in the k-th channel, resulting in:

$$\widehat{\mathcal{M}}_{i,j} = \begin{cases} \mathcal{M}_{i,j}, & S_{i,j} = 1, \\ -inf, & S_{i,j} = 0, \end{cases} \tag{1}$$

where $\mathcal{M}_{i,j}$ and $S_{i,j}$ denote the values at position (i, j) of $\mathcal{M}$ and S, respectively. $\widehat{\mathcal{M}}$ represents the rectified attention score map of k-th channel. For the padding text concept, the S is a binary map with all 1 values. The newly obtained $\widehat{\mathcal{M}}$ thus replaces $\mathcal{M}$ as the attention map.

If using RCA, the cross-attention operation tends to match the given layout, while its ability for general text semantics with a whole sentence will be weaken. To address this issue, we transform the RCA operation to the task-aware rectified cross-attention (TRCA) operation. The idea of TRCA is very simple, *i.e.*, if the condition is layout, we keep the original RCA operation; otherwise, if the condition is instruction, we set all text concepts as padding text concept, *i.e.*, all S are 1-value binary maps. In this way, TRCA can merge both layout and instruction formats, which is beneficial to affect only pixels in the region when given the layout condition and well understand the whole text information when give the instruction condition.

3.2.2 Task-aware gated embeddings

With TRCA, we already achieve a unified image generation pipeline, where both layout and instruction are supported as the inputs. However, the model may encounter difficulty to well learn these two formats, *i.e.*, especially when the image have few text semantics, only small attention maps are modified, which is easy confuse the model learning with that with the instruction input.

To steer the image generation toward a correct direction, we append task-aware gated embeddings (TGE) into the model. Following [45], we also use a unique learned task-aware embedding for the layout and the instruction inputs, serving as input gates to distinguish the input types. TGE is optimized during the model training jointly with the model weights. We integrate TGE into two positions. One is to add it to the timestep embeddings. Another one is to concatenate it with the text embeddings as the inputs of the cross-attention module in the U-Net. The optimization is as:

$$\min_{\theta, k\in\{0,1\}} \mathbb{E}_{y,\epsilon,t}\left[\|\epsilon - \epsilon_\theta(z_t, t, E(c_I), c_T, v_k)\|_2^2\right] \tag{2}$$

where $\epsilon \in N(0, 1)$ is the noise added by the diffusion process. ϵ_θ is the parameters of U-Net. z_t is the noisy latent of the encoded image latent z , where the noise level is related to the timestep $t \in T$. $y = (c_T, c_I, x, i)$ is a quadruplet of text prompt, input image, target image and the index of input format. $v_k, k \in \{0, 1\}$ represents the input format, where v_0 is for the layout input and v_1 is for the instruction input.

In the inference stage, we designate the v_k according to the input format we use. This is different from [45] that predicts the task index with a fine-tuned a large language model, which is simpler, faster and more accurate.

Discussion. The most relative works with ours are FreestyleNet [53], Gen2Det [46], and Emu-Edit [45]. [53, 46] are layout-based methods, and [53] is based on the panoramic segmentatic layouts and [46] is based on bounding boxes. Different from them, we use semantic masks as the layouts, since we target on improving object detection. Meanwhile, although our TRCA is inspired by [53], they are different since TRCA is controled by TGE to choose one of input formats. Therefore, [53] can be seen as a special case of our Uni-SD. [45] also use a task embeddings, where the "task" means different editing tasks that all use instructions. Differently, our TGE is targeted on totally different two input formats, which needs to handle more difficult tasks. Besides, we do not need to predict which task is in the inference stage. We directly choose one of task embeddings to use.

4 Experiments

4.1 Settings

Datasets. To generate a diffusion model with strong generalization performance, we collect datasets from various fields including Detection, Visual Question Answering, and Referring Expression Comprehension. The datasets include COCO [29], LVIS [17], Visual Genome [26] and VGPhraseCut [1].

COCO is a widely-used dataset for object detection and segmentation tasks, with over 200,000 images across 80 categories. LVIS is a long tailed dataset with 1203 classes utilizing the same images as COCO which contains only 80 classes. The long tailed nature of LVIS makes it suitable to verify the use of synthetic data for the rare categories. Visual Genome provides more detailed annotations,

Table 1: Comparisons on COCO with existing methods using Centernet2 architecture. We report Box AP (AP_b) and Mask AP (AP_m).

Method	AP_b	AP_m
Baseline	46.00	39.80
+ Copy-Paste [16]	46.40	39.80
+ X-Paste [58]	46.60	39.90
+ Gen2Det	**47.18**	40.44
+ Uni-SD (ours)	47.06	**40.60**

Table 2: Comparisons on LVIS with existing methods using the Centernet2 architecture. We report Box and Mask AP (AP) and AP for rare/common/frequent classes (AP_r/AP_c/AP_f). We implement synthetic images from [5] using the same instructions with those in our Uni-SD.

Method	Box				Mask			
	AP	AP_r	AP_c	AP_f	AP	AP_r	AP_c	AP_f
Baseline	33.80	20.84	32.84	40.58	29.98	18.36	29.64	35.46
+ InstructPix2Pix [5]	24.88	17.40	24.00	29.16	22.13	15.49	**21.60**	25.64
+ Copy-Paste [16]	34.31	21.19	34.32	40.07	30.29	19.97	**30.43**	34.67
+ X-Paste [58]	34.34	21.05	**33.86**	**40.71**	30.18	18.77	30.11	35.28
+ Gen2Det [46]	34.70	23.78	33.71	40.61	30.82	21.24	30.32	**35.59**
+ Uni-SD (ours)	**34.97**	**25.73**	33.69	40.47	**31.15**	**24.04**	30.21	35.32

including object relationships and attributes, across 100,000+ images. VGPhraseCut is collected based on Visual Genome, including 77,262 images and 345,486 phrase-region pairs.

In the training phase, we use COCO, LVIS, Visual Genome for instruct task and COCO, LVIS, VGPhraseCut for inpaint task. Conversely, In the context of downstream tasks, to ensure fairness, we exclusively utilize COCO or LVIS as the data source to synthetic data.

Implementation details. For training Uni-SD, we use SD-v1.5 [38] as pretrained model and finetune on the aboved training sets for 30 epoch. During inference, we set the diffusion steps N to 50 by default. For synthetic data of instruct task and inpaint task, we don't use any filtering strategy as data post-processing. For object detection and instance segmentation, we follow the settings of X-Paste [58] to run 4× scripts with a batch size of 8 per GPU and a total iteration of 90,000. We use CenterNet2 [59] with ResNet-50 [18] pretrained on the ImageNet dataset with 22,000 categories as the detector. The codebase is built upon Detectron2 [51] and X-Paste [58] . All experiments are run on 8× Tesla V100 32G. The training is about 7 days for training Uni-SD model and 20h for object detection with 16 workers.

4.2 Quantitative evaluation

Comparison on object detection. We compare our Uni-SD to the existing works [16, 58, 46] on the detection and segmentation AP scores with CenterNet2 architecture. Synthetic images are integrated into original images for the detection training.

Table 1 shows that using synthetic images from Uni-SD for the model training, the performance improve by 1.06 and 0.80 on Box AP and Mask AP, suggesting that synthetic images from our Uni-SD are truly useful to offer significant improvements on the object detection and instance segmentation tasks. Meanwhile, our Uni-SD stands out as one of the most effective method, achieving the best performance on Mask AP and comparable Box AP compared to previous state-of-the-art Gen2Det [46] and XPaste [58]. Meanwhile, instead of only supporting layout-formatted inputs as [58, 46], our Uni-SD are more flexible to control what objects to be generated using instructions. This is more practical and is beneficial for improving long-tailed classes as the following LVIS results show.

Comparison on LVIS. We thus evaluate the effectiveness of Uni-SD on the LVIS benchmark [17] in Table 2. LVIS separate classes into frequent, common and rare classes according to how often the category appears in the dataset. Compared to the layout-based methods [58, 46] that only support

Table 3: Quantitative evaluation on the actual image editing tasks with instruction-formatted inputs.

Input format	Method	L1 ↓	L2 ↓	CLIP-I ↑	CLIP-T ↑	DINO ↑
Instruction	InstructPix2Pix [5]	0.302	0.311	0.863	0.227	0.780
	MagicBrush [57]	0.238	0.259	0.900	0.225	0.841
	MGIE [14]	0.325	0.421	0.861	**0.228**	0.783
	Uni-SD (ours)	**0.195**	**0.161**	**0.955**	0.221	**0.970**
Layout	FreestyleNet [53]	0.946	1.448	0.762	-	0.633
	Uni-SD (ours)	**0.355**	**0.413**	**0.891**	-	**0.865**

Table 4: Comparisons on COCO and LVIS with our Uni-SD with and without task-aware gated embeddings (TGE). The detector is Centernet2. We report Box AP (AP_b) and Mask AP (AP_m) for COCO and Box AP (AP) and Box AP rare (AP_r) for LVIS.

Method	COCO		LVIS	
	AP_b	AP_m	AP	AP_r
Baseline	46.00	39.80	33.80	20.84
+ Uni-SD w/o TGE	47.02	40.43	33.98	20.51
+ Uni-SD	**47.06**	**40.60**	**34.97**	**25.73**

masks or bounding boxes inputs, our unified pipeline Uni-SD achieves better performance on overall classes and rare classes, and comparable results on both common and frequent classes. Specially, our method appears a great improvement of 1.17 and 4.89 on Box AP and Box rare AP over baseline model. Compared to [46], our Uni-SD has 1.95 and 2.80 improvement on Box and Mask rare AP. It shows that generated images from our methods have positive effectiveness on object detection and segmentation task, especially on long-tailed settings.

Comparison with image-editing baselines. Existing methods [5, 57, 14, 53, 45] can also perform instruction-based or layout-based image editing. We thus compare our model with several popular image-editing methods, including InstructPix2Pix [5], MagicBrush [57], MGIE [14] and FreestyleNet [53]. We use two main measures: pixel-level and feature-level measurements. For the pixel-level measure, we use L1 and L2 distance metrics to measure pixel-wise modification between the ground-truth target images and the generated images from the COCO dataset. For the feature-level measure, we use CLIP [35] and DINO [7] image encoders. We calculate CLIP-I and DINO to evaluate the similarity between image features of the ground-truth target images and the generated images extracted by CLIP and DINO, respectively. The CLIP-T metric evaluates feature similarity between image and text using text input "add a <obj>", which is only used for the instruction-based input.

We randomly select 2000 images from COCO for evaluation. Table 3 shows that our Uni-SD outperforms other methods across most evaluation metrics during the instruction-based methods. Notably, CLIP-T metric has a bias towards only specifying on objects in the text rather than the entire content of the generated image, resulting in sightly lower performance in Uni-SD than others. In particular, our Uni-SD outperforms MGIE [14] by 0.183 on DINO metric, showing that our method can better follow instructions. As for the layout-formatted inputs, the table shows that our method outperforms FreestyleNet [53] by large margins. The underlying reason is that [53] depends on the panoramic segmentation layouts, where the background will be changed (as show in Figure **??**). Therefore, the synthetic images from [53] is largely different from the original images, especially on the pixel values. Instead, our Uni-SD can keep the background unchanged and only modify the objects, creating more realistic images.

4.3 Qualitative evaluation

Visual comparison with Uni-SD with other methods. For better illustrating the effectiveness of our Uni-SD, we compare the visualization of the editing results between our Uni-SD and other image editing methods. We choose one layout-based method, *i.e.*, FreestyleNet [53] and two instruction-based methods, including InstructPix2Pix [5] and MagicBrush [24] as they are relative to ours.

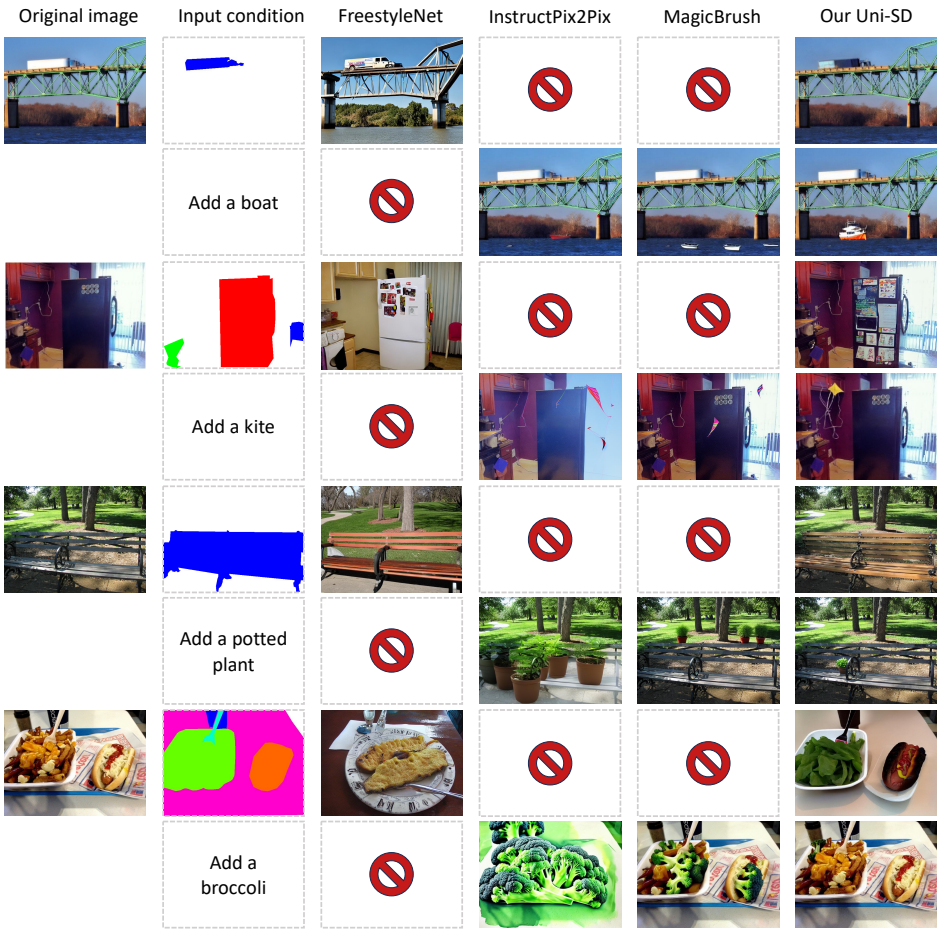


Figure 3: Comparisons of Uni-SD and other image editing methods on generation results with various instructions and layouts. Our Uni-SD has benefits of supporting both layout- and instruction-formatted inputs and generating more realistic images while keep consistency with original images. (Best view in color and zoom in)

In Figure 3 we can see that our Uni-SD is able to generate images according to both layout and instruction input formats. Meanwhile, our method is maintains the background consistency while providing diversity in generalization. For example, in the third row, we can generate a kite in a rational position and the appearance is better than other methods. Simultaneously, our Uni-SD can also change the character of the refrigerator, while keeping other contexts in the image unchanged. There are similar observations in other images, *e.g.*, the added potted plant from our Uni-SD is more realistic and rational than InstructPix2Pix [5] and MagicBrush [24]. And the trees in the background are not modified when changing the diversity of the bench, which FreestyleNet [53] will also modify the background. It clearly show that our method can not only generate realistic images with both two kinds of input formats, but also can keep the consistency of the original image.

4.4 Ablation study

Effectiveness of the task-aware gated embeddings (TGE). To make our Uni-SD well understand both layout and instruction formatted inputs, we design task-aware gated embeddings (TGE) for each input format. In Table 4, we compare the performance of object detection when Uni-SD use TGE or not. It shows that using TGE brings consistent improvement on COCO and LVIS. On LVIS, TGE brings large improvement with 0.98 / 5.22 Box AP / rare AP compared to without using TGE, clearly showing the effectiveness of TGE.

Sampling ratio of the layout- and instruction-formatted inputs. Given generated synthetic images, it is critical to consider how to use these additional images. [58, 46] make effort to balance

Table 5: Ablation on the sampling ratio of realistic images (noted as "real") and synthetic images from our Uni-SD (noted as "syn"). All experiments are conducted on COCO using Centernet2 detector.

Sampling ratio (%)	Box		Mask	
	AP	AP_{50}	AP	AP_{50}
0 (only real)	46.00	-	39.80	-
10 (real:syn=9:1)	46.51	65.41	40.35	62.45
20 (real:syn=8:2)	**47.06**	**66.01**	**40.60**	**62.66**
30 (real:syn=7:3)	47.00	65.99	40.45	62.56
40 (real:syn=6:4)	46.61	65.57	40.06	62.20

Table 6: Ablation on the data proportion between instruction- and layout-formatted synthetic images from our Uni-SD. All experiments are conducted on COCO using Centernet2 detector.

Data proportion	Box		Mask	
instruct : layout	AP	AP_{50}	AP	AP_{50}
1 : 0	46.51	65.40	40.24	62.22
0 : 1	46.72	65.71	40.33	62.58
1 : 1	**47.06**	**66.01**	**40.60**	**62.66**

Table 7: Study on what kinds of synthetic data are useful for the object detection. We conduct all experiements on LVIS to evaluate the effectiveness of frequent, common and rare classes. The detector is Centernet2.

Class	Box				Mask			
	AP	AP_r	AP_c	AP_f	AP	AP_r	AP_c	AP_f
Baseline	33.80	20.84	32.84	40.58	29.98	18.36	29.64	35.46
+ only frequent	34.53	21.45	34.01	40.86	30.59	19.97	30.16	35.59
+ only common	34.59	22.45	34.09	40.49	30.66	20.25	30.68	35.22
+ only rare	34.67	25.35	33.21	40.41	30.76	22.58	29.93	35.29
+ all (Uni-SD)	**34.97**	**25.73**	33.69	40.47	**31.15**	**24.04**	30.21	35.32

the amount of realistic data and synthetic data using the sampling ratio. Here, we follow [58, 46] to explore this factor. Table 5 shows that the relative best sampling ratio is 20% in our method, *i.e.*, the proportion of real data and generated data is 4:1.

Besides, as our Uni-SD can generate two types of synthetic data based on the layout- and instruction-formatted input, we also study the data proportion between these two synthetic data. From Table 6, the best data proportion is 1:1, showing that two types of synthetic images similar equal usefulness.

What kinds of synthetic data are useful for object detection? Reviewing the LVIS dataset, it divides the object categories into three hierarchy, including frequent, common and rare classes, according to the instances' quantity. This motivates to explore what kinds of synthetic data can bring improvements on the object detection. In this setting, we fix to use the same layout-based synthetic images, while changing to use different classes for generation on the instruction-based inputs. Table 7 shows that all kinds of synthetic images are useful to improve the performance, especially the rare classes. It show that our method is good to address the long-tailed problem. If there are too few instances in the dataset, our Uni-SD is a good choice to enrich their diversity.

5 Conclusion

In this paper, we introduces Uni-SD, an innovative image generation pipeline that effectively integrates the strengths of both layout-based and instruction-based methods. Our Uni-SD not only can use flexible and user-friendly instruction inputs but also maintain the precision and diversity with layout inputs in the generated images. The proposed task-aware rectified cross-attention (TRCA) and task-aware gated embeddings (TGE) are able to enhance the model's ability to adapt and respond to different inputs, ensuring that generated data is accurate and contextually appropriate. Experiments on object detection, image-editing tasks and visualization show the effectiveness of our method.

Limitations and social impact. Our findings are based on public datasets and model architectures, which may limit their generalizability. Future work can explore a wider range of datasets to validate our results. Our method can be used to improve currect perception tasks, including object detection, segmentation, classification, which will improve the accuracy in the real application, especially on the long-tailed situation.

References

[1] Vgphrasecut. `https://github.com/ChenyunWu/PhraseCutDataset`, 2019.

[2] O. Avrahami, D. Lischinski, and O. Fried. Blended diffusion for text-driven editing of natural images. In *CVPR*, 2022.

[3] S. Azizi, S. Kornblith, C. Saharia, M. Norouzi, and D. J. Fleet. Synthetic data from diffusion models improves imagenet classification. *arXiv preprint arXiv:2304.08466*, 2023.

[4] 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. *Computer Science. https://cdn. openai. com/papers/dall-e-3. pdf*, 2023.

[5] T. Brooks, A. Holynski, and A. A. Efros. Instructpix2pix: Learning to follow image editing instructions. In *CVPR*, 2023.

[6] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *NeurIPS*, 2020.

[7] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, pages 9650–9660, 2021.

[8] S. Chai, L. Zhuang, and F. Yan. Layoutdm: Transformer-based diffusion model for layout generation. In *CVPR*, pages 18349–18358, 2023.

[9] J. Chen, R. Zhang, Y. Zhou, and C. Chen. Towards aligned layout generation via diffusion model with aesthetic constraints. *arXiv preprint arXiv:2402.04754*, 2024.

[10] X. Chen, L. Huang, Y. Liu, Y. Shen, D. Zhao, and H. Zhao. Anydoor: Zero-shot object-level image customization. *arXiv preprint arXiv:2307.09481*, 2023.

[11] G. Couairon, J. Verbeek, H. Schwenk, and M. Cord. Diffedit: Diffusion-based semantic image editing with mask guidance. *arXiv preprint arXiv:2210.11427*, 2022.

[12] N. Dvornik, J. Mairal, and C. Schmid. Modeling visual context is key to augmenting object detection datasets. In *ECCV*, 2018.

[13] D. Dwibedi, I. Misra, and M. Hebert. Cut, paste and learn: Surprisingly easy synthesis for instance detection. In *ICCV*, 2017.

[14] T.-J. Fu, W. Hu, X. Du, W. Y. Wang, Y. Yang, and Z. Gan. Guiding instruction-based image editing via multimodal large language models. *arXiv preprint arXiv:2309.17102*, 2023.

[15] R. Gal, Y. Alaluf, Y. Atzmon, O. Patashnik, A. H. Bermano, G. Chechik, and D. Cohen-Or. An image is worth one word: Personalizing text-to-image generation using textual inversion. *arXiv preprint arXiv:2208.01618*, 2022.

[16] G. Ghiasi, Y. Cui, A. Srinivas, R. Qian, T.-Y. Lin, E. D. Cubuk, Q. V. Le, and B. Zoph. Simple copy-paste is a strong data augmentation method for instance segmentation. In *CVPR*, 2021.

[17] A. Gupta, P. Dollar, and R. Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *CVPR*, 2019.

[18] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[19] R. He, S. Sun, X. Yu, C. Xue, W. Zhang, P. Torr, S. Bai, and X. Qi. Is synthetic data from generative models ready for image recognition? *arXiv preprint arXiv:2210.07574*, 2022.

[20] A. Hertz, R. Mokady, J. Tenenbaum, K. Aberman, Y. Pritch, and D. Cohen-Or. Prompt-to-prompt image editing with cross attention control. *arXiv preprint arXiv:2208.01626*, 2022.

[21] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *NeurIPS*, 2020.

[22] D. A. Hudson and L. Zitnick. Generative adversarial transformers. In *ICML*, pages 4487–4499. PMLR, 2021.

[23] J. Ji, R. Zhao, and M. Lei. Latent diffusion transformer for point cloud generation. *The Visual Computer*, pages 1–15, 2024.

[24] B. Kawar, S. Zada, O. Lang, O. Tov, H. Chang, T. Dekel, I. Mosseri, and M. Irani. Imagic: Text-based real image editing with diffusion models. In *CVPR*, 2023.

[25] J. Y. Koh, D. Fried, and R. R. Salakhutdinov. Generating images with multimodal language models. *NeurIPS*, 2024.

[26] R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, J. Kravitz, S. Chen, Y. Kalantidis, L.-J. Li, D. A. Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *IJCV*, 2017.

[27] X. Li, Y. Ren, X. Jin, C. Lan, X. Wang, W. Zeng, X. Wang, and Z. Chen. Diffusion models for image restoration and enhancement–a comprehensive survey. *arXiv preprint arXiv:2308.09388*, 2023.

[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 *CVPR*, 2023.

[29] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.

[30] S. Liu, Z. Zeng, T. Ren, F. Li, H. Zhang, J. Yang, C. Li, J. Yang, H. Su, J. Zhu, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*, 2023.

[31] S. Lu, Y. Liu, and A. W.-K. Kong. Tf-icon: Diffusion-based training-free cross-domain image composition. In *ICCV*, 2023.

[32] A. Nichol, P. Dhariwal, A. Ramesh, P. Shyam, P. Mishkin, B. McGrew, I. Sutskever, and M. Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021.

[33] OpenAI. Chatgpt. `https://openai.com/chatgpt`, 2024.

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

[35] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

[36] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

[37] A. Ramesh, M. Pavlov, G. Goh, S. Gray, C. Voss, A. Radford, M. Chen, and I. Sutskever. Zero-shot text-to-image generation. In *ICML*, 2021.

[38] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022.

[39] N. Ruiz, Y. Li, V. Jampani, Y. Pritch, M. Rubinstein, and K. Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *CVPR*, 2023.

[40] C. Saharia, W. Chan, S. Saxena, L. Li, J. Whang, E. L. Denton, K. Ghasemipour, R. Gontijo Lopes, B. Karagol Ayan, T. Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *NeurIPS*, 2022.

[41] M. B. Sariyildiz, K. Alahari, D. Larlus, and Y. Kalantidis. Fake it till you make it: Learning transferable representations from synthetic imagenet clones. In *CVPR*, 2023.

[42] S. Sheynin, A. Polyak, U. Singer, Y. Kirstain, A. Zohar, O. Ashual, D. Parikh, and Y. Taigman. Emu edit: Precise image editing via recognition and generation tasks. *arXiv preprint arXiv:2311.10089*, 2023.

[43] J. Song, C. Meng, and S. Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[44] Q. Sun, Y. Cui, X. Zhang, F. Zhang, Q. Yu, Z. Luo, Y. Wang, Y. Rao, J. Liu, T. Huang, et al. Generative multimodal models are in-context learners. *arXiv preprint arXiv:2312.13286*, 2023.

[45] Q. Sun, Q. Yu, Y. Cui, F. Zhang, X. Zhang, Y. Wang, H. Gao, J. Liu, T. Huang, and X. Wang. Generative pretraining in multimodality. *arXiv preprint arXiv:2307.05222*, 2023.

[46] S. Suri, F. Xiao, A. Sinha, S. C. Culatana, R. Krishnamoorthi, C. Zhu, and A. Shrivastava. Gen2det: Generate to detect. *arXiv preprint arXiv:2312.04566*, 2023.

[47] Z. Tang, C. Wu, J. Li, and N. Duan. Layoutnuwa: Revealing the hidden layout expertise of large language models. In *ICLR*, 2024.

[48] Y. Tian, L. Fan, P. Isola, H. Chang, and D. Krishnan. Stablerep: Synthetic images from text-to-image models make strong visual representation learners. *NeurIPS*, 36, 2024.

[49] N. Tumanyan, M. Geyer, S. Bagon, and T. Dekel. Plug-and-play diffusion features for text-driven image-to-image translation. In *CVPR*, 2023.

[50] N. Wasserman, N. Rotstein, R. Ganz, and R. Kimmel. Paint by inpaint: Learning to add image objects by removing them first. *arXiv preprint arXiv:2404.18212*, 2024.

[51] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick. Detectron2. `https://github.com/facebookresearch/detectron2`, 2019.

[52] G. Xiao, T. Yin, W. T. Freeman, F. Durand, and S. Han. Fastcomposer: Tuning-free multi-subject image generation with localized attention. *arXiv preprint arXiv:2305.10431*, 2023.

[53] H. Xue, Z. Huang, Q. Sun, L. Song, and W. Zhang. Freestyle layout-to-image synthesis. In *CVPR*, pages 14256–14266, 2023.

[54] B. Yang, S. Gu, B. Zhang, T. Zhang, X. Chen, X. Sun, D. Chen, and F. Wen. Paint by example: Exemplar-based image editing with diffusion models. In *CVPR*, 2023.

[55] C.-H. Yeh, T.-Y. Cheng, H.-Y. Hsieh, C.-E. Lin, Y. Ma, A. Markham, N. Trigoni, H. Kung, and Y. Chen. Gen4gen: Generative data pipeline for generative multi-concept composition. *arXiv preprint arXiv:2402.15504*, 2024.

[56] T. Yu, R. Feng, R. Feng, J. Liu, X. Jin, W. Zeng, and Z. Chen. Inpaint anything: Segment anything meets image inpainting. *arXiv preprint arXiv:2304.06790*, 2023.

[57] K. Zhang, L. Mo, W. Chen, H. Sun, and Y. Su. Magicbrush: A manually annotated dataset for instruction-guided image editing. *NeurIPS*, 2024.

[58] H. Zhao, D. Sheng, J. Bao, D. Chen, D. Chen, F. Wen, L. Yuan, C. Liu, W. Zhou, Q. Chu, et al. X-paste: Revisiting scalable copy-paste for instance segmentation using clip and stablediffusion. In *ICML*, 2023.

[59] X. Zhou, V. Koltun, and P. Krähenbühl. Probabilistic two-stage detection. *arXiv preprint arXiv:2103.07461*, 2021.

A More details of datasets

Designing train datasets for inpaint task For inpaint task, we use the instance annotations and images of COCO, Lvis, and VGPhraseCut. For one image, We use the prompt $< c_1\ c_2\ c_3\ ...\ c_n >$ where c_i is the class name for corresponding instance and n is the number of instance in the image.

Designing Instructions of Removal-Dataset for instruct task To improve the user experience and usability, we enhance the instruction's format using ChatGPT [33] to generate multiple templates, such as "Put the <object> into the figure." Leveraging ChatGPT, we also create various templates for instructions, such as "Add a <object>." or "Put the <object> into the figure." Additionally, we generate more complex sentences like "Integrate a <obj> into the scene for enhancement," as well as interrogative forms like "Would you consider inserting a <obj> into the composition?" . Note that for generating synthetic data for quantitative analysis on detection, we only use the simplest instruction, i.e., "Add a <object>," to generate images for fast application.

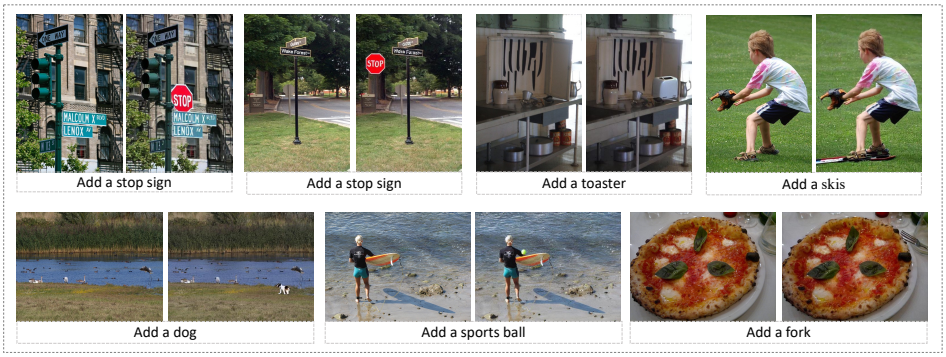
Dataset Merging Strategy Initially, we attempt to directly mix the synthetic datasets with the original one using a weighted sampler. However, since we do not predict masks for the synthetic data, the mask loss is only computed on the real data. This situation causes instability in gradient updating of mask losses, as it requires multiplying a binary mask to disable calculation on generated samples without the ground truth mask. As a result, similar to Gen2Det [46], we perform batch sampling instead, which samples the entire batch with either original data or synthetic data controlled by a threshold. This approach avoids additional processing on the mask loss when the current batch is sampled from synthetic data

Codebase We implement our experiments upon the official code of FreestyleNet [53], InstructPix2Pix [5], Stable Diffusion [38], XPaste [58] . The respective links and licenses are detailed in Tab. 8.

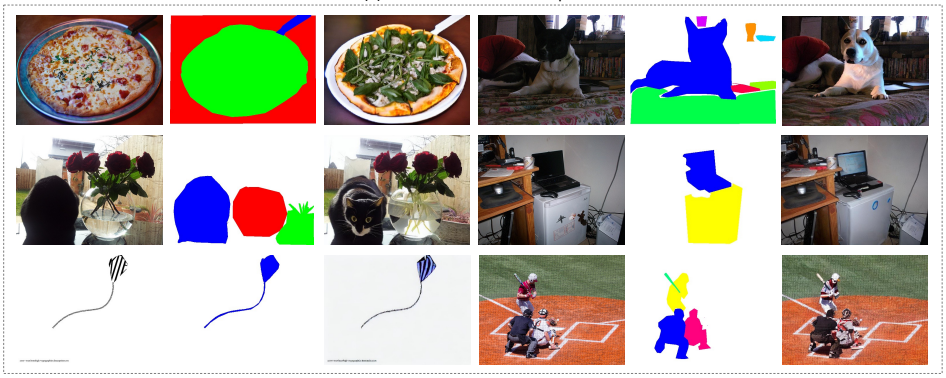
Table 8: **Code links and licenses.**

Method	Link	License
FreestyleNet [53]	https://github.com/essunny310/FreestyleNet	MIT License
InstructPix2Pix [5]	https://github.com/timothybrooks/instruct-pix2pix	Open RAIL++-M
Stable Diffusion [38]	https://github.com/CompVis/stable-diffusion	Open RAIL++-M
XPaste [58]	https://github.com/yoctta/XPaste	Apache 2.0
GroundingDINO [30]	https://github.com/IDEA-Research/GroundingDINO	Apache 2.0

More visualization As our Uni-SD supports both instruction and layout input formats, here we provide more visualizations in Figure 4 for further showing the qualitative effectiveness of our method. The Figure shows that our method can well response the instruction input as well as the layout input.



(a) Instruction-formatted input



(b) Layout-formatted input

Figure 4: Visualization of synthetic images from COCO dataset with (a) instruction-formatted input and (b) layout-formatted input from our Uni-SD. For each pair or triplet images, the first image is the original image, the last image is the generated image, and the center image in the triplet is the layout. (Best view in color and zoom in)