SceneComposer: Any-Level Semantic Image Synthesis Supplementary Material

Yu Zeng[1], Zhe Lin[2], Jianming Zhang[2], Qing Liu[2], John Collomosse[2], Jason Kuen[2], Vishal M. Patel[1]
{yzeng22,vpatel36}@jhu.edu, {zlin,jianmzha,qingl,collomos,kuen}@adobe.com
[1]Johns Hopkins University [2]Adobe Research

A. Concepts Interpolation

With the proposed text feature map representation, we can create mixtures of concepts using overlapping shapes with different text descriptions. We can change the importance of different concepts in the generated image by adjusting their weights. Fig. A shows example images generated from mixtures of concepts. We can see the gradual transition from the first concept to the second as the weights change.

Figure A. Images generated from mixtures of two concepts. The weights for the second concepts gradually increase from left to right.

B. Image Editing/Inpainting Results

Given an image and a mask, the proposed method can be applied for image editing/inpainting by sampling the unmasked regions using the information in the original image, as in [6]. More specifically, at each diffusion step, the masked regions are filled with the model output and the unmasked regions are from the noise corrupted original image. Fig. B shows the editing/inpainting results. For this application, we can also specify the layout and the precision levels. When the precision

level > 0, our method provides an intuitive image editing tool that allows users to edit by painting with a semantic brush. When the precision level is set to 0, our method can inpaint a masked region (the frog example in Fig. B).

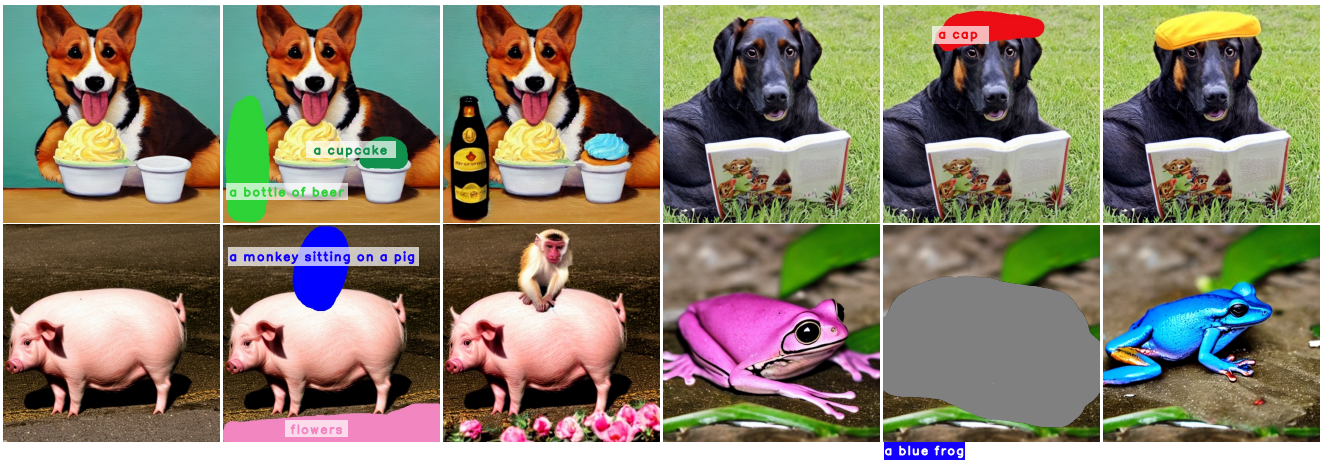
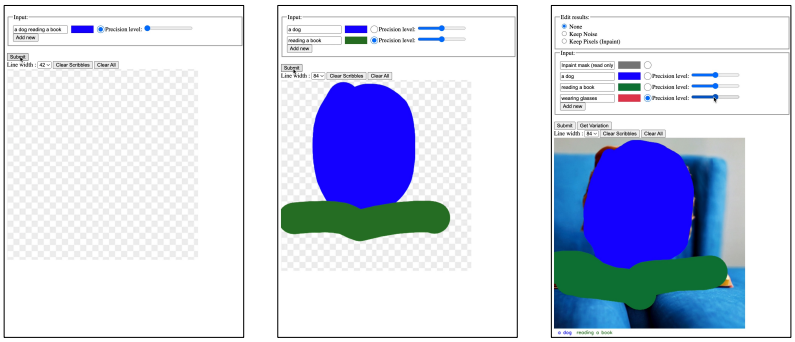


Image Layout Result Image Layout Result

Figure B. Image inpainting/editing results.

C. Interactive Image Synthesis

We build an interactive image synthesis application using the proposed framework. Fig. C shows the user interface at different statuses. At the initial status (Fig. C (a)), users define semantic concepts with free-from texts and may adjust the precision levels optionally. Then they can draw a layout on the blank canvas, or choose the lowest precision level and directly click submit (Fig. C (b)). The generated images will be depicted on the canvas to allow further refinements/editing (Fig. C (b)). At this stage, the users can choose to keep the noise from which the images are sampled to get an image of a similar style, or choose the inpainting mode to edit the results. More details regarding the interactive image synthesis application can be found in the demo video.



(a) (b) (c)

Figure C. The user interface for interactive image synthesis.

D. Results with Mixed Precision Levels

In the proposed framework, each semantic region can be assigned with an individual precision level to enable a more flexible control mechanism. In the paper, we have shown the use case of mixing a 0-th level style indicator with other semantic regions of high levels. Here, we demonstrate more use cases of mixed precision levels in Fig. D. For each input layout, we sample five images to reflect the difference in the strength of control of different precision levels. We can see that the object shapes in regions of lower precision shows more variations across different samples (*e.g.* the bears in the first row and the skateboards in the second row) whereas those with higher precision are closer to each other.

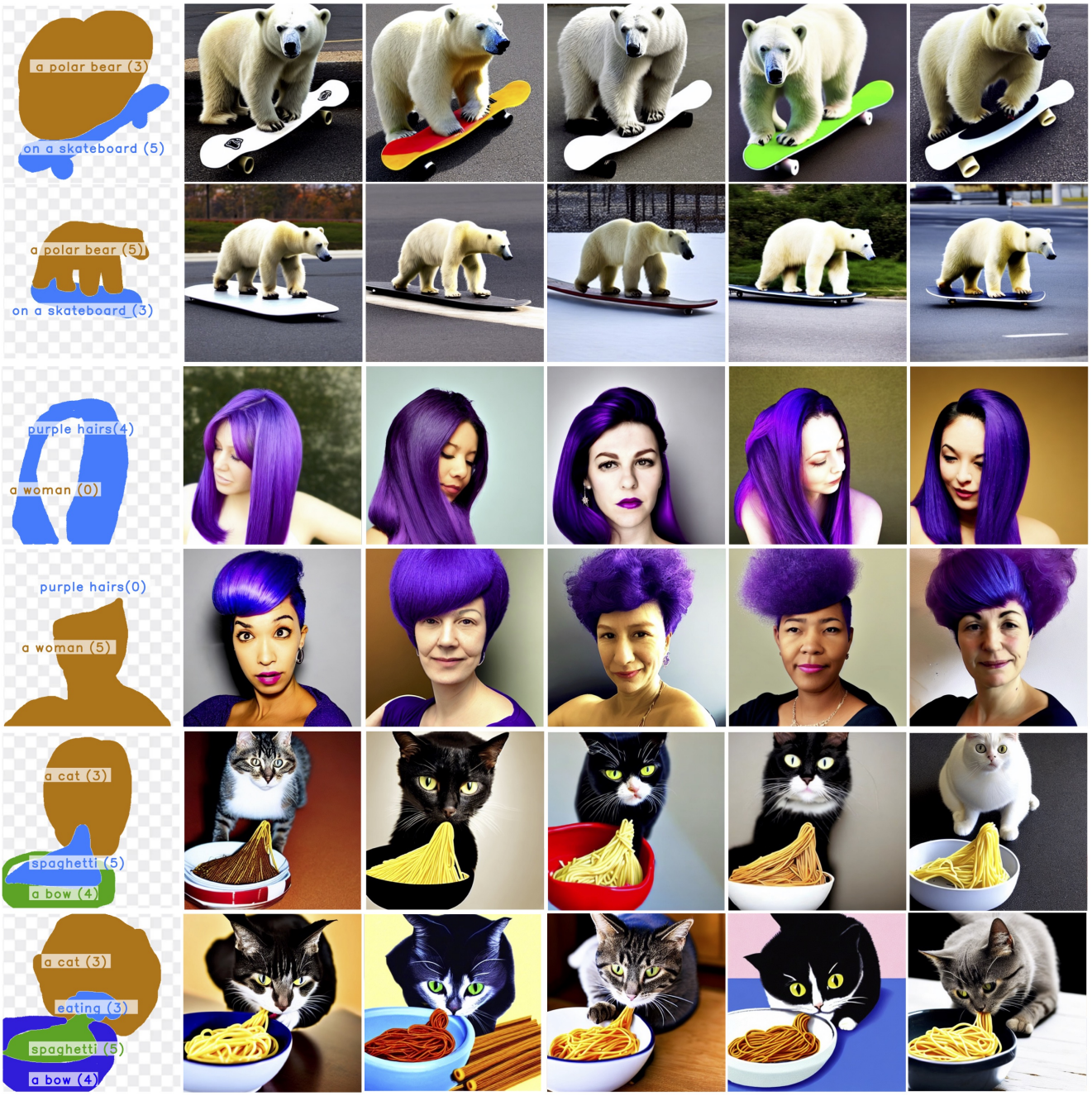


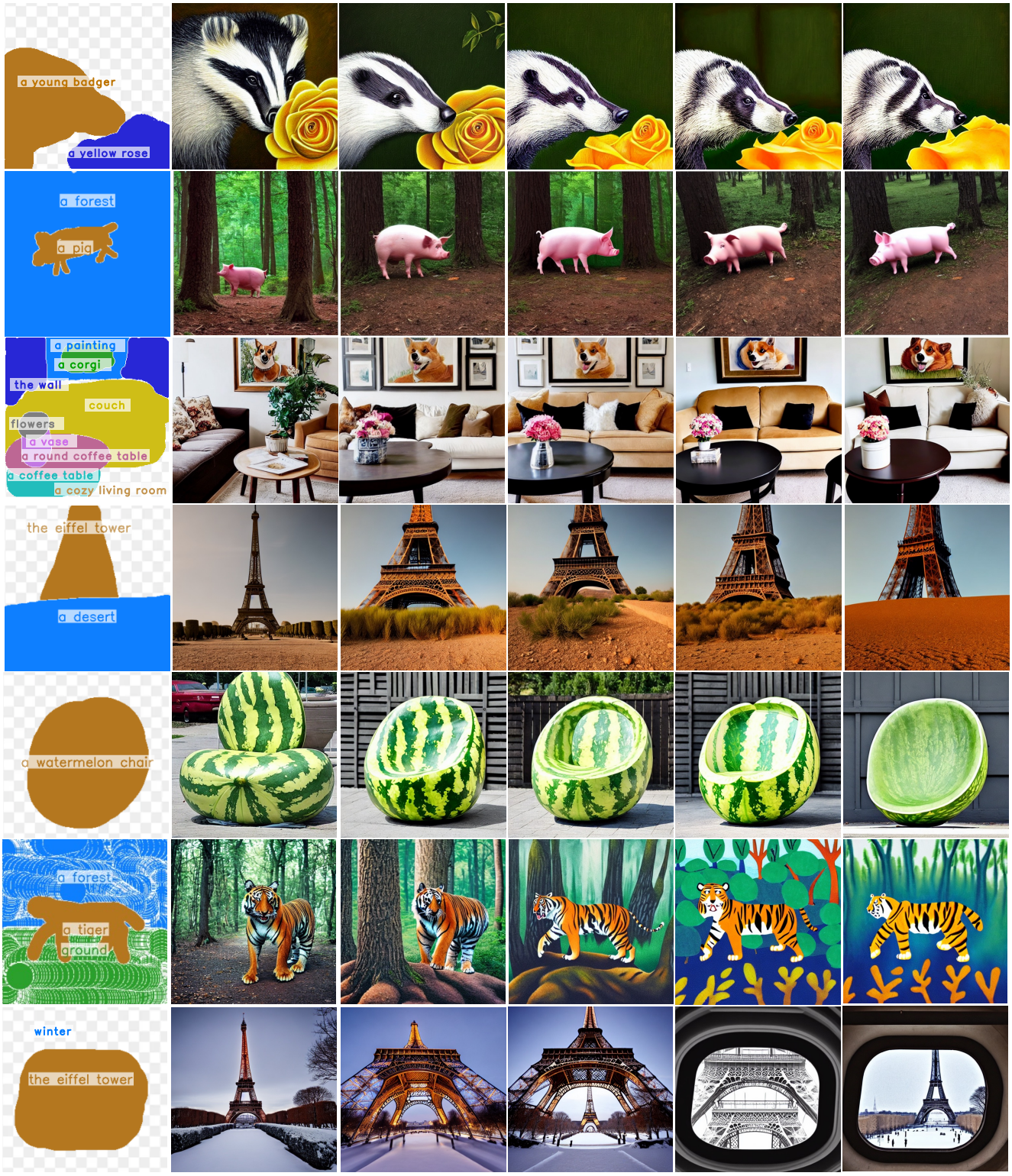
Figure D. Images generated from layouts with mixed precision levels. Number in the brackets indicate the precision levels. Regions of 0-th levels are not shown.

E. Additional Samples with Varying Precision Levels

Fig. E shows additional samples generated from layouts at different precision levels. When a coarse layout is assigned with a high precision leve, the model will still try to fit in the specified shapes by cartoonizing the images (*e.g*. the last two results of the sixth row) or generating extra structures (*e.g*. the extra photo frames in the last example).

F. Additional Analysis

Image synthesis from bounding box layout. As indicated in Sec 4.2 of the paper, our method can generate images from bounding box layouts using the coarsest precision level. Here we report the quantitative comparison results on the COCO-Stuff



Input Level 0 Level 3 Level 4 Level 5 Level 6

Figure E. Results generated using different precision levels from the same layout and starting from the same noise.

validation set against the existing bounding box based methods in Table A. We process the data follow the settings of [3, 11, 13]. Our method can achieve comparable FID to existing methods without being trained on bounding box layout, and outperforms them after being finetuned on COCO-Stuff training set with bounding box layout.

Table A. Quantitative comparison with bounding box based layout-to-image synthesis models on COCO-Stuff.

	Method	FID ↓	zero-shot FID↓
64×64	LostGAN-V1 [10]	34.31	-
	OC-GAN [12]	33.1	-
	Layout2Im [13]	44.19	-
	Ours	30.27	38.25
256×256	LostGAN-V2 [11]	42.55	-
	OC-GAN [12]	41.65	-
	LDM [8]	40.91	-
	Ours	35.69	41.74

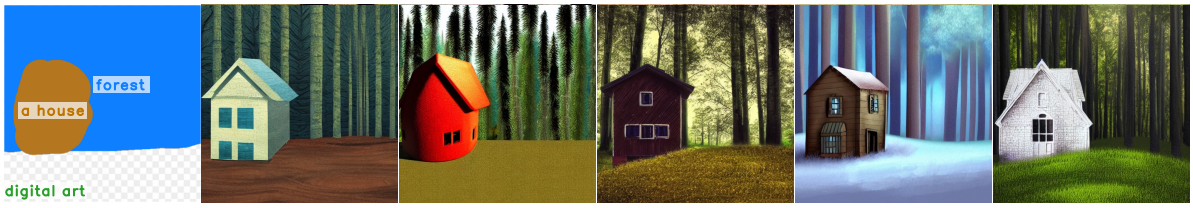


Figure F. Diverse samples. Levels of non-empty masks are set to 3.

Classifier-free guidance. At inference time, we apply the classifier-free guidance based on the unconditional estimates from an empty text feature pyramid of the repeating $\boldsymbol{f}(\oslash)$. Here we explore the use of a partially empty text feature pyramid, *i.e.* only apply the classifier-free guidance with selected levels set to $\boldsymbol{f}(\oslash)$. The results are reported in Table B. Using classifier-free guidance with all levels yields the best performance (The last row of Table B).

Table B. Results with classifier-free guidance applied at selected levels of the text feature pyramid. *Level* indicates which levels to drop out when computing the unconditional estimates.

Level	CLIP Score ↑	SS Score ↑
6	.230	.708
5,6	.229	.714
4,5,6	.229	.718
3,4,5,6	.229	.733
0,3,4,5,6	.261	.736

Sensitivity to the order of concatenation. We use the embedding of the concatenation of all regional text descriptions as a 1×1 feature map at the 0-th level of the text feature pyramid. We found the overall performance is not sensitive to the concatenation order, as indicated in Table C.

Table C. Results with different concatenation orders. Ours: the class labels of different regions are concatenated according to the index in the COCO-Stuff dataset. Ours-permute: the class labels are concatenated in random order.

Method	FID ↓	mIOU↑
Ours	17.20	23.01
Ours-permute	17.21	22.76

Diverse results. For the same layout, our method can generate diverse results by sampling from different noise. We show the diverse samples from the same input in Fig. F.

Effect of verbs masks for action generation. Fig. G shows example results illustrating the effect of verb masks. The verb masks can enhance the emphasis on the actions (see the comparison of the first two examples to the third example in the top row). We can also use verb masks to control the object orientation (the first and second dog examples), shapes, and background (the bear examples) complimentary to the object masks.

Figure G. Example results demonstrating the effect of verb masks.

SS Score compared to mIOU. SS Score is defined as the average cosine similarity between the entries of the text feature maps of the input layout and the layout predicted from the generated image. Compared to mIOU, the SS Score is more suitable for evaluating images generated from open-domain layouts, where the same object can correspond to multiple similar labels, *e.g.* a white mug, a plain mug, and a white-colored mug. Table D compares the mIOU and SS Score at different precision levels on OpenLayout-COCO, which show a similar trend.

Table D. SS score and mIOU on OpenLayout-COCO of different levels.

Prec. Level	0	3	4	5	6
SS Score↑	.572	.685	.716	.729	.736
mIOU↑	0.09	15.59	17.83	19.26	19.03

Human evaluation. Table. E reports human evaluation results of our method and the best performing methods in terms of automatic metrics (SD in Table 3 and SPADE in Table 4). The results show that our method is favored by human subjects.

Table E. User evaluation results on 100 random samples. We ask 10 users to select the better one from two generated images shown in a random order.

Dataset	COCO		COCO-stuff	
Method	SD	Ours	SPADE	Ours
Preference rate (%) ↑	39.8	60.2	11.4	88.6

G. Implementation Details

Following the practices in [7–9], we adopt the UNet architecture in [1] for the 64×64 diffusion models. For the super-resolution model in experiments with the pixel-space diffusion, we use the same architecture as in [1]. We use the decoder from [8] for the latent-space diffusion model. The hyper-parameters for the 64×64 model are summarized as follows.

```
image_size: 64
in_channels: 4 #3 for the pixel-space diffusion
out_channels: 4 #3 for the pixel-space diffusion
model_channels: 320 #192 for the pixel-space diffusion
attention_resolutions: [ 32, 16, 8 ]
num_res_blocks: 2
channel_mult: [ 1, 2, 4, 4 ] #[1, 2, 3, 4] for the pixel-space diffusion
```

Fig. 5 in the paper illustrates the overall architecture of the modified UNet. Here we show the architecture of each block in Fig. H. The original architecture in [1] is shown on the left side as a reference.

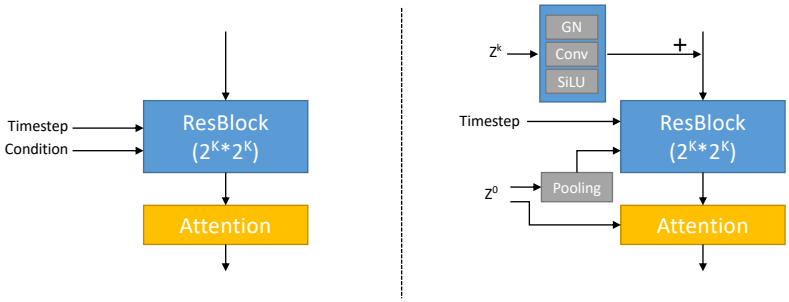


Figure H. The architecture of a block in the original UNet in [1] (left) and our modified UNet (right). GN: Group Normalization; SiLU: Sigmoid Linear Unit.

We use the cosine noise schedule for diffusion. We uniformly sample from 1000 diffusion steps in training. At inference time, we use the PLMS sampler [5] and sample for 100 diffusion steps. We set the classifier-free guidance scale to 3 for evaluation in the text-to-image generation setting and 2 for segmentation-to-image generation. The samples for visualization are generated with the guidance scale set to 7.5. The precision level is set to 3 by default if not specified. It takes 32 seconds to generate a batch of 16 samples on a A100 GPU.

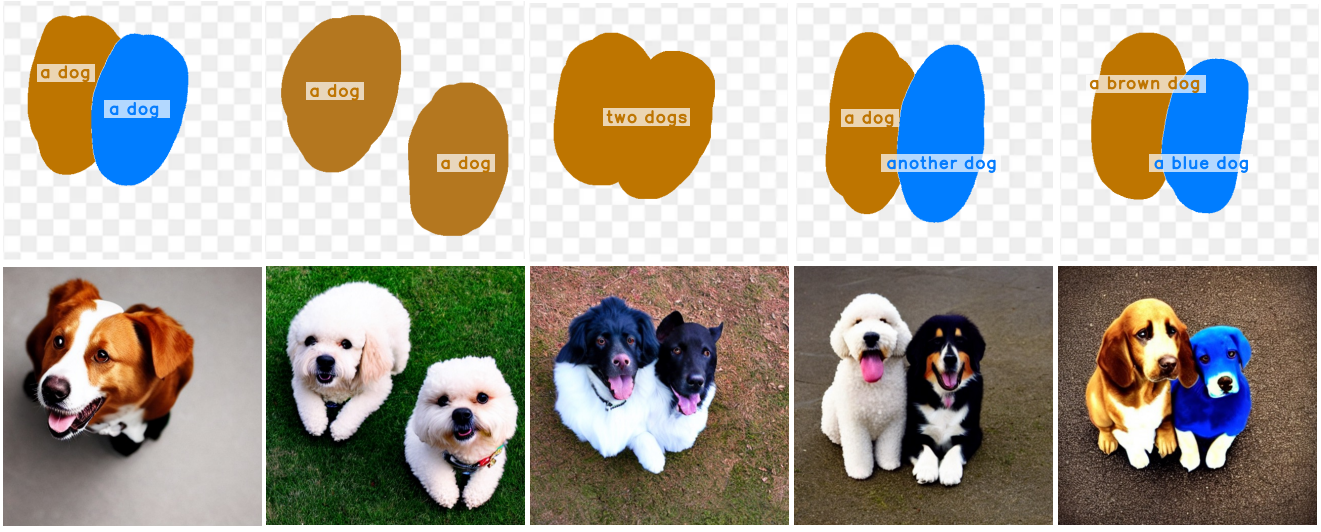
We annotate the layouts for training using text-based object detection [4] and segmentation [2]. More specifically, for an image with a caption, we use [4] to generate bounding boxes for entities that appear in the caption. Then we compute the similarity of each entity to the categories in COCO. We use the mask estimation branch of [4] to obtain a segmentation mask if the maximum similarity is over 0.3 and keep the mask of the bounding box otherwise.

H. Limitations and Future Work

Since the text feature maps are not instance-aware, different instances of the same category or with the same text descriptions correspond to the same features. Therefore, the model cannot separate adjacent instances if their text descriptions are the same, as shown in Fig. I (a). There are several ways to generate multiple instances with our model:

- Clearly define the instance boundary in the layout (Fig. I (b)).
- Specify the number of instances (Fig. I (c)).
- Emphasize multiple instances in text descriptions (Fig. I (d)).
- Specify the attributes of different (Fig. I (e)).

However, these are still not very intuitive and require additional user effort. It can be a direction for future work to incorporate instance information in the proposed framework.



(a) (b) (c) (d) (e)

Figure I. The text feature maps are not instance-aware. We need to specify the number of instances or attributes to separate instances of the same semantic category.

References

[1] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Conference on Neural Information Processing Systems*, 34:8780–8794, 2021. 6, 7

[2] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *International Conference on Computer Vision*, pages 2961–2969, 2017. 7

[3] Justin Johnson, Agrim Gupta, and Li Fei-Fei. Image generation from scene graphs. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1219–1228, 2018. 4

[4] Liunian Harold Li*, Pengchuan Zhang*, Haotian Zhang*, Jianwei Yang, Chunyuan Li, Yiwu Zhong, Lijuan Wang, Lu Yuan, Lei Zhang, Jenq-Neng Hwang, Kai-Wei Chang, and Jianfeng Gao. Grounded language-image pre-training. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 7

[5] Luping Liu, Yi Ren, Zhijie Lin, and Zhou Zhao. Pseudo numerical methods for diffusion models on manifolds. In *International Conference on Learning and Representation*, 2022. 7

[6] Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 1

[7] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021. 6

[8] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 5, 6

[9] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S Sara Mahdavi, Rapha Gontijo Lopes, et al. Photorealistic text-to-image diffusion models with deep language understanding. *arXiv preprint arXiv:2205.11487*, 2022. 6

[10] Wei Sun and Tianfu Wu. Image synthesis from reconfigurable layout and style. In *International Conference on Computer Vision*, pages 10531–10540, 2019. 5

[11] Wei Sun and Tianfu Wu. Learning layout and style reconfigurable gans for controllable image synthesis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(9):5070–5087, 2021. 4, 5

[12] Tristan Sylvain, Pengchuan Zhang, Yoshua Bengio, R Devon Hjelm, and Shikhar Sharma. Object-centric image generation from layouts. In *he AAAI Conference on Artificial Intelligence*, 2021. 5

[13] Bo Zhao, Lili Meng, Weidong Yin, and Leonid Sigal. Image generation from layout. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 8584–8593, 2019. 4, 5