Grounded Text-to-Image Synthesis with Attention Refocusing Supplementary Material

Quynh Phung Songwei Ge Jia-Bin Huang
University of Maryland, College Park
{quynhpt, songweig,jbhuang}@umd.edu

I. Comparing with attention-based optimizing methods

We highlight the differences in four **Grounded**: We use explicit layout to mitigate guidance ambiguity caused by multiple objects of the same category or complex spatial relationships. **Optimization-based**: We optimize the attention losses by iteratively refining the *noised sample*, which is more effective than modifying the *attention maps* like in DenseDiffusion. **Outside-box**: We regularize the attention scores outside the grounding regions. This constraint avoids generating objects in unrelated regions. **Self-attention**: We also optimize self-attention maps following the input layout. Self-attention loss helps minimize feature leakage. **Attention-refocusing** applied the above four techniques, leading to improvement compared to other attention-based methods in HRS and DrawBench (Table 1 and Table 8)

	Grounded	Optimization-based	Outside box	Self-attention
Attend-Excite	×	✓	×	×
A-STAR	×	✓	×	×
LayoutGuidance	✓	✓	×	×
DenseDiffusion	✓	×	✓	✓
Boxdiff	✓	✓	✓	×
Ours	✓	✓	✓	✓

II. Implementation of grounded text-to-image models

II.1. Data set and metrices

Dataset

The HRS dataset contains various prompts divided into three main categories: 1) accuracy, 2) robustness, and 3) generalization. Our method focuses on *accuracy improvement*, including four main categories: *spatial relationship, color, size, and counting*. Each prompt in the dataset is tagged with the object's name and corresponding labels intended for evaluation. For example, in spatial relationships, the labels include objects and their relative positions, such as "on the left" or "on the right". The prompts for each category counting/spatial/size/color are $3,000/1,002/501/501$. Depending on the number of objects and their relationship, we label the difficulty level of each prompt as easy, medium, and hard with roughly the same amount.

The DrawBench dataset consists of 39 prompts about *Counting* and *Positional (or spatial relationship)*. Since there are no labels for this benchmark, we manually create the label for each prompt based on the number of objects mentioned and their relationships.

The TIFA benchmark contains $4,000$ prompts in various categories (counting, spatial, food, locations, etc...) and the questions for each prompt, along with their answers. **the dataset provided by DenseDiffusion** It includes about 250 binary masks with corresponding labels and captions, allowing us to evaluate our method's capability in adhering to the provided mask guidances.

Evaluation metrics. We use metrics in the HRS to compute the accuracy of individual categories. For counting, the precision, recall, and F1 scores are used to measure text-to-image models. False positive samples happen when the number of generated objects is smaller than the ground truths. In contrast, the false negative objects are counted for the missing objects in the synthesized images. For other categories, we use accuracy as the evaluation metric. Depending on the category, the image is counted as a correct prediction when all detected objects are correct for spatial relationships, color, or size.

For the TIFA benchmark, we assess the alignment between the generated images and input texts using TIFA score. This metric is calculated based on a question-answering model, which uses generated images as input and outputs an answer for the specific question in the dataset. Then, the predicted answer is compared with the corresponding ground truth. For FID, We randomly choose 5k captions and corresponding images from this benchmark to calculate the FID. We use this evaluation to validate that our generation results remain natural compared to the base model.

II.2. Implementation details of CAR and SAR losses

We apply Cross-Attention Refocusing and Self-Attention Refocusing losses on the attention maps of resolution 16×16. All images are generated with 50 steps of denoising. We discuss setting details for optimization during denoising steps, referring to Eq. (7) In terms of τ, in the very early steps ($t = 0$ or $t = 1$), the cross and self-attention maps are unclear yet begin to form the layout. So, we just set the iteration step $\tau = 2$. Then, to make the layout clearer ($t \in \{2, 3, 4\}$), τ is increased to 6 steps, which helps refine the layout if tokens do not attend to the corresponding boxes or are in the wrong boxes. We also apply early stopping to reduce inference time and ensure the quality of generated images. We observe that applying optimization in later steps can lead to quality degradation. Therefore, after the first ten denoising steps, we only update the latent when the tokens do not align with the corresponding boxes or with incorrect ones. The initial step size α is set to 4 in the first five steps, then decreases to 3. The detail of the algorithm can be seen in Algorithm 25

In this paper, we use a Gaussian kernel with filter size 3 $\times$ 3 and a σ value of 0.5 for standard deviation

In terms of four baselines, layout-to-image models: layout-guidance, MultiDiffusion, Attend-and-Excite, GLIGEN, they are set default in their original papers.

II.3. Applying CAR loss for segmentation mask

We also adapt the CAR loss to other layout modalities like depth maps, segmentation masks, and edge maps. Specifically, we always use the converted segmentation masks M_i associated with token i-th to apply our method. Since the segmentation provides a precise object boundary in contrast to the bounding box, we optimize the attention over the entire foreground by taking the average instead of the maximum. The foreground loss for segmentation masks is:

$$\mathcal{L}_{FG} = \frac{1}{q} \sum_{i \in I} \frac{\sum(1 - (A_i^t \cdot M_i))}{\sum M_i} \tag{8}$$

Similarly, the background loss for segmentation maps is:

$$\mathcal{L}_{BG} = \frac{1}{q} \sum_{i \in I} \frac{\sum A_i^t \cdot (1 - M_i)}{\sum(1 - M_i)} \tag{9}$$

The L_{SAR} is calculated using the formulation for bounding box presented in the main paper. The results of applying our losses to ControlNet are shown on our website (open the file index.html)

Algorithm 1: Denoising step with Attention-Refocusing

Data: A text prompt P, a set of token indices I, each token associates with a set of bounding box B_i , a timestep t, a set of iterations for refinement $\{t_1, \ldots, t_k\}$, the threshold T, and a trained Stable Diffusion model SD.

Result: latent x_{t-1} for the next timestep

$A^t, S^t \leftarrow SD(z, P, t)$
$A^t \leftarrow \text{Softmax}(A_t - \text{sot}())$
for $i \in I$ **do**
  $A_i^t \leftarrow A_{t[:,:,i]}$
  $A_i^t \leftarrow \text{Gaussian}(A_i^t)$
  $L_i^{t,FG} \leftarrow 1 - \max(A_i^t \cdot \text{Mask}(B_i))$
  $L_i^{t,BG} \leftarrow \max(A_i^t \cdot (1 - \text{Mask}(B_i)))$
  $L_{i,CAR} \leftarrow L_i^{FG} + L_i^{BG}$
  for $p \in Mask(B_i)$ **do**
    $L_p = \sum_{p \in B_i}(\text{Average}(S_p^t \cdot (1 - \text{Mask}(B_i))$
  end
  $L_{i,SAR} = \sum_p (L_p)$
end
$L_{CAR} \leftarrow \sum_i (L_{i,CAR})$
$L_{SAR} \leftarrow \sum_i (L_{i,SAR})$
$L \leftarrow L_{CAR} + L_{SAR}$
$\hat{x}_t \leftarrow x_t - \alpha_t \nabla_{x_t} L$
if $t \in \{t_1, \ldots, t_k\}$ **then**
  if $L > 1 - T$ **then**
    $x_t \leftarrow \hat{x}_t$
    Go to Step 1
  end
end
$x_{t-1} \leftarrow SD(\hat{x}_t, P, t)$
return x_{t-1}

III. Layout generation

III.1. Full prompt for GPT-4

Our full prompt mainly includes the three components:
Instruction specifies the task and defines the output format. This instruction helps GPT-4 perform better in layout generation tasks.
In-context exemplars are used further to enhance the model's capacity for the task. We supplement user prompts with multiple examples for the best context understanding. This also helps the model output the desired form of bounding boxes and their corresponding labels.
User prompt is appended to the instruction and the supporting examples. Then, the model completes the chat conversation from the user prompt and returns the layout in the defined form.

Once the user provides a prompt (user prompt), it will be added to the defined and fixed text to create a full prompt shown in Table 7. Then, the GPT-4 API completes the chat and returns the box coordinates of the corresponding objects.

The comparison of our two-stage text-to-image models

with single-stage one (Stable diffusion) and several appealing results generated from our framework can be found on our website (open the file index.html).

III.2. Comparison of four language models

Metrices for large language models evaluation We evaluate large language models in three metrics:

- Format: whether or not the model returns the correct format of grounded information, including four coordinates for each box and its label.
- Validness: all generated boxes are satisfied with the size and box constraints, eg. the coordinate box is $\{x_1, y_1, x_2, y_2\}$ then $512 \geq x_1, x_2, y_1, y_2 \geq 0$, $x_1 \leq x_2$ and $y_1 \leq y_2$,
- Correctness: the generated grounding information should follow the text prompts. For example, in terms of counting, the quantity of generated boxes should match the number of objects mentioned in the input prompt. In spatial and size categories, we asses the relations and relative size of generated boxes. Meanwhile, in color, we verify if the correct colors are returned for each object in the grounding text.

The comparison of four language models The results are shown in Fig. 11. GPT-4 is capable of reasoning implicit object relationships. For instance, in the first prompt, a squirrel with a leather racket, GPT-4 can place the leather racket box centrally within the squirrel box, unlike GPT-3, Llama 2, and Llama1, which miss the spatial composition. However, the GPT-4 still struggles with some extremely hard prompts. For prompts with too many objects, GPT-4 can generate the correct number of objects but with small boxes.

IV. Additional quantitative results

The Table 8 shows results of our methods in the drawbench . Our proposed losses demonstrate a comparable performance boost to HRS. By integrating our losses, we compare favorably or comparatively against baselines in the counting procedure. Moreover, our losses substantially improve the accuracy of the spatial category. We compare the time inference of our losses and other free-training methods in Table 10. Our losses are compatible with Attend-and-excite , even more effective than Layout-guidance in speed.

V. Additional applications

V.1. Instructing text-to-image by chatGPT

We also propose a novel capability enabled by our framework, where users can utilize chatGPT to instruct text-to-image. In other words, after generating the initial layout and image, we instruct chatGPT to modify the layout, leading to an updated image. This iterative capability allows users to synthesize desired images through consecutive adjustments. As shown in Fig. 12, a user wants to generate an appealing image of "a hot air balloon flying over a field of four giant marshmallows". At first, the generated layout was not satisfying, so the user asked chatGPT to shift the balloon layout to the right and then add the sun to its left. Such language-based refinement ability is difficult for traditional text-to-image models to offer.

VI. Ablation study

We compute the accuracy, FID, and inference time (s/image) in the spatial category in the HRS benchmark to ablate iteration and step-size.

Ablation study of iteration: The ablation study of iteration can be seen in 11. With a fixed step-size of 4, increasing iteration, FID first decreases and then increases, inference time keeps increasing, whereas accuracy significantly improves. The sweet spot lies around 3 to 4 iterations.

Ablation study of step-size: The ablation study of step-size can be seen in the 12. We set iteration at 4 to explore the effect of step-size. The FID remains stable with step size from 0 to 7 and increases from 7 to 15. Importantly, accuracy increases significantly from a step size of 0 to 4, then not improved.

Table 7. The full prompt for gpt4 api.

Role	Content
Instruction	System: "You are ChatGPT-4, a large language model trained by OpenAI. Your goal is to assist users by providing helpful and relevant information. In this context, you are expected to generate specific coordinate box locations for objects in a description, considering their relative sizes and positions and the number of objects. The box coordinates should be in the order (left, top, right, bottom). The size of the image is 512*512."
In-context examples	User: "Provide box coordinates for an image with a cat in the middle of a car and a chair. Make the size of the boxes as big as possible." Assistant: "cat: (245, 176, 345, 336); car: (10, 128, 230, 384); chair: (353, 224, 498, 350)" User : "Provide box coordinates for an image with three cats on the field." Assistant: "cat: (51, 82, 399, 279);cat: (288, 128, 472, 299); cat: (27, 355, 418, 494)"
User prompt	User : "Provide the Provide box coordinates for an image with" + [user prompt]

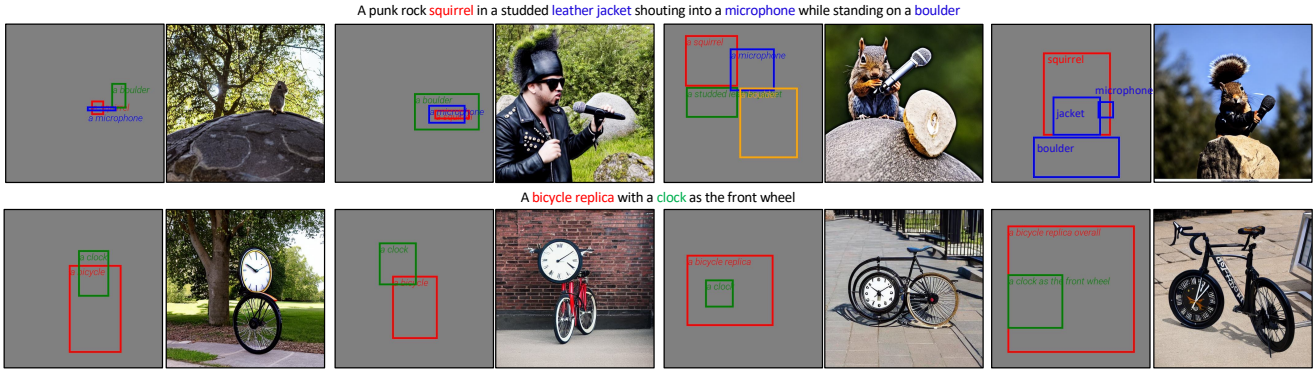

Figure 11. Comparison generated layouts from Llama 1, Llama 2, GPT-3, GPT-4

Table 8. Quantitavie evaluation on the DrawBench benchmark.

Method	CAR & SAR	Counting			Spatial
		Precision ↑	Recall ↑	F1 ↑	Accuracy ↑
Stable Diffusion	×	73.32	70.00	71.55	12.50
	✓	78.53 (+5.2)	73.63 (+3.6)	75.81 (+4.3)	43.50 (+31.0)
Attend-and-excite	×	77.64	74.85	76.20	20.50
	✓	74.06 (-3.6)	77.58 (+2.7)	75.66 (-0.5)	38.00 (+18.0)
Layout-guidance	×	79.15	70.61	74.48	36.50
	✓	78.45 (-0.7)	75.45 (+4.8)	76.82 (+2.3)	52.50 (+16.0)
MultiDiffusion	×	75.37	65.61	69.90	38.00
	✓	84.30 (+8.9)	68.03 (+2.4)	75.20 (+5.3)	54.50 (+16.5)
GLIGEN	×	81.66	80.89	81.18	48.00
	✓	90.28 (+8.6)	86.21 (+5.3)	88.16 (+7.0)	64.00 (+16.0)

Table 9. Our proposed losses improve the baselines in the HRS Counting benchmark.

Method	CAR & SAR	Precision ↑	Recall ↑	F1 ↑
Stable Diffusion	×	71.86	52.19	58.31
	✓	81.56 (+9.7)	51.19 (-1.0)	60.62 (+2.3)
Attend-and-excite	×	73.10	54.79	60.47
	✓	75.94 (+2.8)	56.31 (+1.5)	62.71 (+2.2)
Layout-guidance	×	80.60	45.83	56.22
	✓	78.15 (-2.5)	55.65 (+9.8)	63.01 (+6.8)
MultiDiffusion	×	78.96	45.18	55.18
	✓	83.26 (+4.3)	45.71 (+0.5)	57.37 (+2.2)
GLIGEN	×	78.81	59.44	66.58
	✓	81.25 (+2.4)	59.39 (-0.1)	67.54 (+0.7)

Table 10. Inference time of different methods (s/10 images). AE: Attend-and-Excite, MD: MultiDiffusion, LG: Layout-guidance, Ours: Attention-Refocusing

Stable Diffusion					GLIGEN	
SD	+AE	+MD	+LG	+Ours	GLIGEN	+Ours
54.33	101.67	74.16	111.13	102.97	205.90	279.08

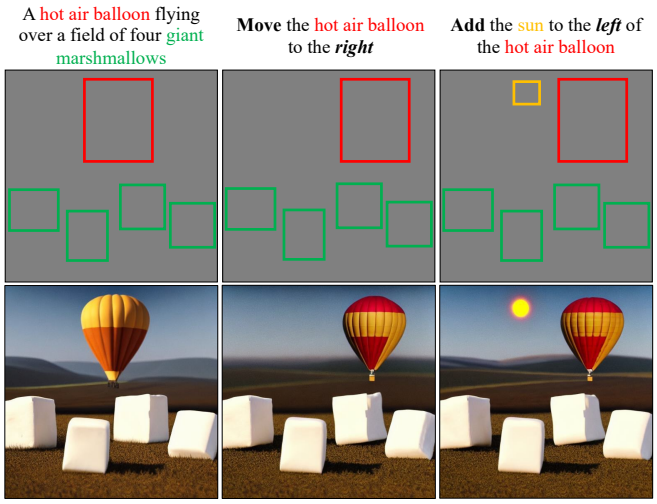

Figure 12. Instruct text-to-image by instructing chatGPT.

Table 11. Ablation study of iteration steps

Iteration	0	3	4	5	6	8	10	15
Acc ↑	33.53	39.22	**41.11**	40.82	41.02	41.00	40.32	36.52
FID ↓	67.71	**67.11**	67.60	67.67	67.39	68.29	68.69	70.45
Time ↓	**20.50**	21.53	22.03	22.41	23.64	25.02	25.95	28.42

Table 12. Ablation study of Step size

Step-size	0	2	3	4	5	7	9	15
Acc ↑	33.53	41.11	40.32	**42.31**	41.91	40.71	41.52	41.91
FID ↓	67.71	67.97	68.08	**67.26**	67.45	67.70	68.89	72.35