Supplementary Material: Layout Free Scene Graph to Image Generation

Rameshwar Mishra
rameshwarm@iiitd.ac.in
A V Subramanyam
subramanyam@iiitd.ac.in

Indraprastha Institute of Infromation Techonology
Delhi, India

1 Experimental Setup

Dataset. We train and evaluate our model on COCO-stuff and Visual genome dataset. We process our data following existing works [1, 3]. After pre-processing, we get 62,565 image-graph pair in training set and 5,506 image-graph pair in validation set of Visual Genome dataset. COCO-stuff has 40,000 and 5,000 image-graph pairs in training and validation sets respectively. We follow [3] to create synthetic scene graphs for COCO-stuff using spatial relationship edges.

Evaluation Metrics. To show effectiveness of our approach we evaluate our model using Inception Score (IS) [6], Frechet Inception Distance (FID) [2], Diversity Score (DS) [9], and Object occurrence ratio (OOR) [8]. IS is a metric commonly used to evaluate the quality and diversity of generated images in generative models. A higher Inception Score indicates better-performing generative models that produce both realistic and diverse images. DS is a measure used to quantify the variety and distinctiveness of generated samples for same input scene graph. FID evaluates the similarity between the distribution of real data and generated data using feature representations extracted from a pre-trained Inception model. OOR is the ratio of the objects detected in the generated image by YOLOv7 [7] with respect to the objects given in the input scene graph. High OOR implies high consistency of generated images with scene graphs.

Training Parameters. We use a pre-trained stable diffusion model [5]. Graph encoder is a standard multi layer graph convolution network taking nodes and edges as input. d_g for graph encoder is 512, we take $\lambda = 0.7$ and $\beta = 0.5$. For reconstruction loss in diffusion, we guide are training with the MSE loss between predicted and added noise. We use Adam optimizer [9] with a learning rate of 1e-6. We fine-tune the Diffusion model for 62,000 iteration and 32,525 iterations for Visual Genome and COCO-stuff datasets respectively,with batch size of 2. Discriminator is a 5 layer MLP, trained for 40 epochs with Adam optimizer.

2 Architectural Details

Our architecture consists of three primary components: the GAN based CLIP alignment (GCA) module, a text-to-image diffusion model, and a graph encoder. This section provides architectural details for these components.

2.1 Diffusion Network

We use Stable Diffusion V1-4 checkpoint [4] as our diffusion model. The hyperparameter values for diffusion model is given in table 1. We employ the DDPM noise scheduler with 1000 diffusion timesteps. To generate 256×256 images, we utilize an input noise latent of size 32×32×4.

Hyperparameter	Considered value
Input noise shape	$32 \times 32 \times 4$
Noise scheduler	DDPM scheduler
Diffusion timesteps	1000
Autoencoder type	KL-regularized
Learning rate scheduler	constant
Unet's CA resolutions	32,16,8

Table 1: Hyperparameter values for the diffusion model. Unet refers to the denoising network of diffusion. CA refers to cross-attention.

Net.	Layer(Input shape)	Output Shape
Object Net.	Linear (512)	512
	ReLU	512
	Linear (512)	512
	ReLU	512
Triplet Net.	Linear (3×512)	512
	ReLU	512
	Linear (512)	512
	ReLU	512

Table 2: Architecture of the graph convolution layer. Object network and triplet networks are joined Parallelly. All layers are sequentially added to create the respective network.

2.2 Graph Encoder

Following previous works [3] we use a graph convolution network to encode our scene graph. Graph encoder consists of 5 graph convolution layers. Table 2 shows the architecture of a single graph convolution layer. This layer consists of two parallel networks, one to predict object embedding and the other to predict triplet embedding.

Table 3 illustrates the comprehensive architecture of our graph encoder. Initially, object labels and relationship labels are fed into a vocabulary-based embedding layer. The input for the triplet network in the graph convolution layer is formed by concatenating the embeddings of the subject (S), relationship (R), and object (O) in a scene graph relationship triplet (S, R, O). 5 graph convolution layers are sequentially added to predict the object and triplet embeddings. In table 3, GraphConv takes two inputs, object embedding of size 512 and 512×3 dimension concatenated input for triplet network. It outputs 512 dimension object and triplet embedding. We apply average pooling to get global object and triplet embedding.

Net	Layer(Input type/shape)	Output shape
Embedding Net.	Object Layer (Label)	512
	Relation Layer(Label)	512
Graph Net.	GraphConv (512,512×3)	512,512
	GraphConv (512,512×3)	512,512
	GraphConv (512,512×3)	512,512
	GraphConv (512,512×3)	512,512
	GraphConv (512,512×3)	512,512
Projection Net.	Avg Pool ($N_O \times 512$)	512
	Avg Pool ($N_T \times 512$)	512
	Linear (2×512)	512

Table 3: Architecture of graph encoder Network. Object layer and relationship layer are two parallel embedding layers. Layers of Graph Network are sequentially connected. Projection net consists of two parallel average pooling layers. Output of these pooling layers is concatenated and fed to a linear layer.

Finally, we project the concatenated global object embedding and triplet embedding to get our 512 dimension graph embedding.

2.3 GAN based CLIP alignment module

This module follows a standard GAN architecture. We consider graph encoder as our generator and it's architecute is given in table 3. Architecture of discriminator is given in the Table 4. We use clip-vit-base-patch32 checkpoint of CLIP to get CLIP features for GCA.

3 Additional results

Qualitative ablation results for GCA. Figure 1 demonstrates that outputs generated with the use of GCA are more consistent with the input scene graph. For instance, in the first row, the model without GCA produces distorted birds, whereas in the second row, incorporating GCA leads to correctly spelled words in the image.

Additional qualitative results of our methodology versus existing approaches. Figure 2 showcases additional results, demonstrating the strong alignment of our

Layer(Input shape)	Output shape
Linear (768)	256
BatchNorm	256
LeakyReLU	256
Dropout	256
Linear (256)	128
BatchNorm	128
LeakyReLU	128
Dropout	128
Linear (128)	1
Sigmoid (1)	1

Table 4: All the layers are sequentially added to create the discriminator network. We use a negative slope of 0.2 for LeakyReLU. Dropout probability is 0.3

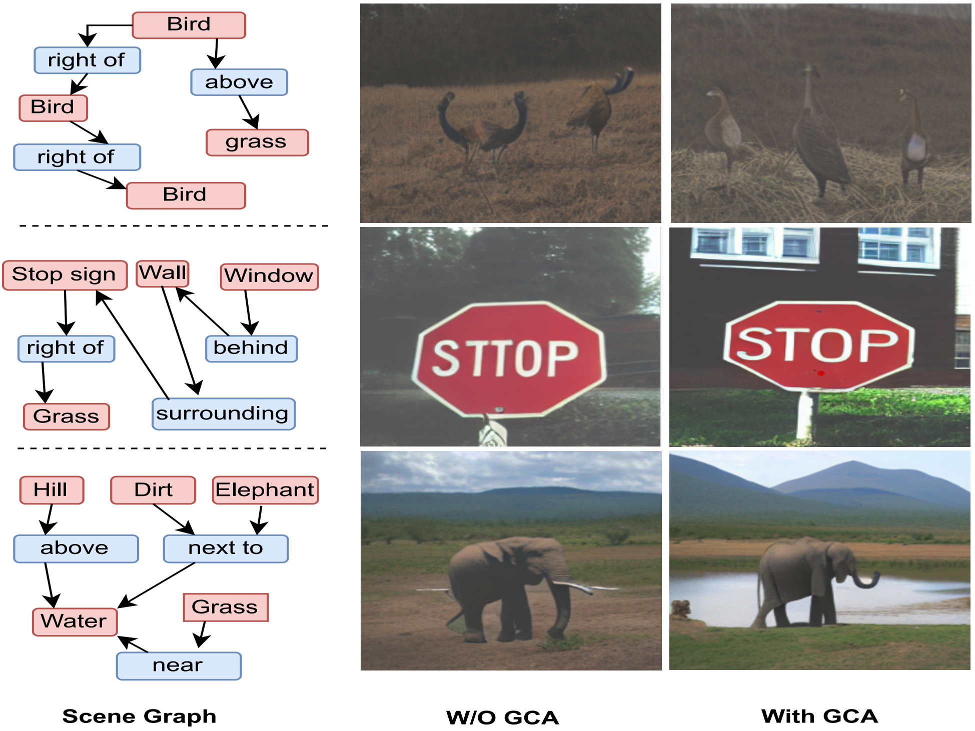

Figure 1: Qualitative results showing the effectiveness of GCA module. GCA refers to GAN based graph alignment. W/O is abbreviation for without. Column 1 contains input scene graphs, while Columns 2 and 3 display results generated without and with the use of GCA, respectively.

method in generating images with the input scene graph. Our model produces diverse images. For instance, in row 3, both Canonical and SGTransformer generate outputs with a blue train structure, aligning with the ground truth containing a blue train. In contrast, our model generates an image featuring a red train. While our image maintains consistency with the input scene graph, it also introduces distinct elements, setting it apart from the original.

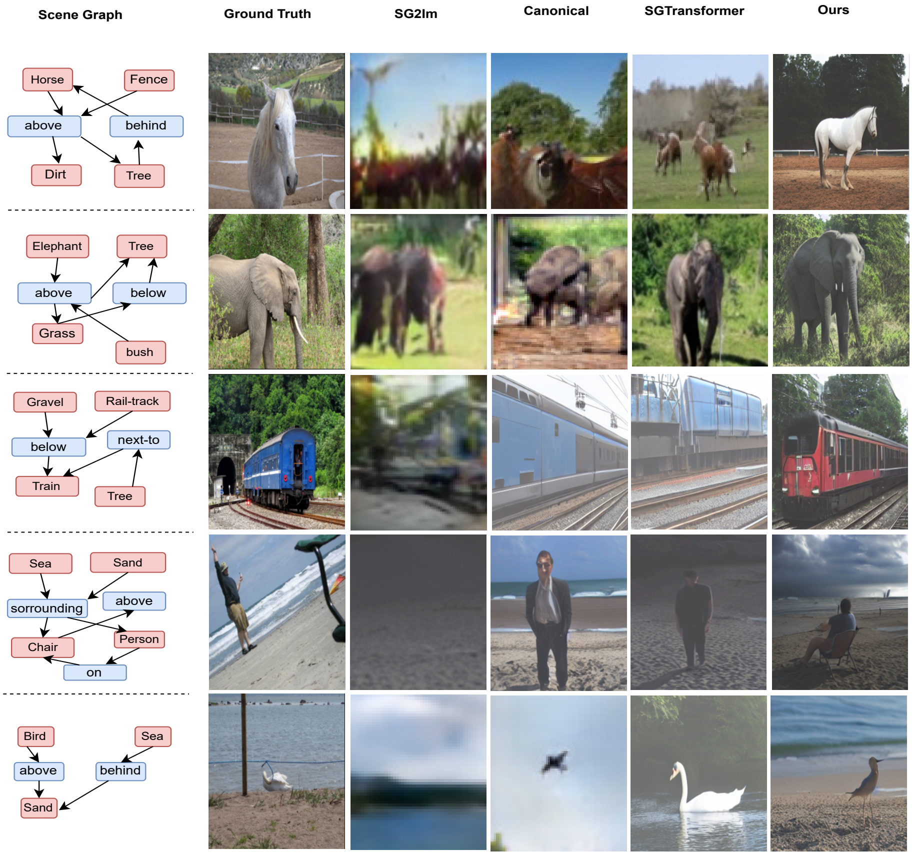

Figure 2: Sample images generated using different existing methods for comparison. It can be seen that our model generates high quality yet diverse images. Reference scene graphs are slightly perturbed to check effectiveness of each method.

References

[1] Roei Herzig, Amir Bar, Huijuan Xu, Gal Chechik, Trevor Darrell, and Amir Globerson. Learning canonical representations for scene graph to image generation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVI 16*, pages 210–227. Springer, 2020.

[2] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

[3] Justin Johnson, Agrim Gupta, and Li Fei-Fei. Image generation from scene graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1219–1228, 2018.

[4] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10684–10695, June 2022.

[5] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[6] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *Advances in neural information processing systems*, 29, 2016.

[7] Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7464–7475, 2023.

[8] Yangkang Zhang, Chenye Meng, Zejian Li, Pei Chen, Guang Yang, Changyuan Yang, and Lingyun Sun. Learning object consistency and interaction in image generation from scene graphs. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, pages 1731–1739, 2023.

[9] Zijun Zhang. Improved adam optimizer for deep neural networks. In *2018 IEEE/ACM 26th international symposium on quality of service (IWQoS)*, pages 1–2. Ieee, 2018.