Compositional Visual Generation via LoRA-Enhanced Stable Diffusion with Depth Conditioning

Naixin Lyu & Phyllis Chen & Tong Zeng
10-423/623 Generative AI Course Project
{naixinly, phyllis2, tongzeng}@andrew.cmu.edu

December 12, 2025

1 Introduction

Text-to-image diffusion models can synthesize visually compelling images from natural language, yet they remain unreliable on compositional prompts that require simultaneously grounding multiple objects, their attributes, and their spatial relationships. Typical failure modes include incorrect attribute binding (e.g., swapping colors between objects), omitted entities, and inconsistent spatial arrangements. In this work, we target these errors with parameter-efficient adaptation rather than inference-time model composition: we combine Low-Rank Adaptation (LoRA) with depth-conditioned Stable Diffusion trained on LAION-SG's structured scene-graph supervision, and evaluate improvements on T2I-CompBench across attribute binding, object relations, numeracy, and complex composition.

2 Dataset and Task

2.1 Dataset

We use the LAION-SG dataset [Li et al., 2024], which contains approximately 20 million image-text pairs from LAION-5B augmented with scene graph annotations. For the use of this project, 100,000 images-text pair were extracted from the dataset for fine-tuning and evaluation. Each image includes structured representations of objects, attributes (color, size, material), and relationships (spatial relations like "left of," "above," "inside"). This structured supervision provides explicit signals for compositional understanding.

2.2 Task

Our goal is to fine-tune Stable Diffusion to handle complex compositional prompts. The model must correctly capture:

- **Attribute binding**: Assign correct attributes to objects (e.g., "red car and blue truck" should not produce a blue car)
- **Object relationships**: Generate correct spatial arrangements (e.g., "cat on table")
- **Multiple object generation**: Include all mentioned objects
- **Spatial reasoning**: Handle directional relationships (left/right, above/below, inside/outside)

2.3 Evaluation Metrics

We evaluate using T2I-CompBench [Huang et al., 2023], which provides compositional quality metrics:

- **Attribute Binding Accuracy**: Measures correct attribute-object associations. This is our primary metric as attribute binding is a major failure mode.
- **Object Relationship Score**: Evaluates spatial and relational accuracy between objects.
- **Generative Numeracy**: Assesses correct generation of specified object counts.
- **Complex Composition Score**: Evaluates performance on multi-element compositional prompts.

T2I-CompBench uses detection-based metrics for quantitative evaluation. If budget permits, we will also leverage MLLM-based evaluation (GPT-4V or open-source alternatives) for nuanced compositional assessment. Otherwise, we will rely on CLIP-based verification, which correlates well with human judgments for attribute binding tasks.

3 Related Work

Our project targets a recurring limitation of text-to-image diffusion models: faithfully following *compositional* prompts that specify multiple objects, attributes, and spatial relationships. Prior work improves compositionality through three main directions—(i) inference-time composition of multiple models, (ii) training-free attention control, and (iii) explicit spatial/structured conditioning—while recent parameter-efficient tuning methods make it feasible to adapt large diffusion backbones under limited compute. Below we synthesize these threads and position our approach: a

single-model Stable Diffusion fine-tuned with *scene-graph-derived supervision* and *depth conditioning* via *LoRA*.

3.1 Composable Diffusion

Composable diffusion methods build complex generations by combining independently trained models at inference. The energy-based formulation of compositional diffusion composes score functions through conjunction/negation, enabling Boolean-style concept composition without retraining for every combination [Liu et al., 2022]. Expert-based approaches such as eDiff-I train specialized diffusion experts (e.g., style/content/layout) and optionally distill them into a unified model, but still incur substantial training cost due to the expert ensemble stage [Balaji et al., 2022]. These methods motivate our design goal: instead of paying multi-model overhead at training or inference, we aim to *internalize* compositional reasoning into a *single* SD backbone using lightweight fine-tuning.

3.2 Attention Manipulation

A complementary line of work improves compositional prompt following without updating model weights by manipulating attention during sampling. Structured Diffusion Guidance decomposes prompts and steers cross-attention maps to strengthen attribute–object alignment [Feng et al., 2023]. Attend-and-Excite addresses catastrophic neglect (missing entities) by optimizing latent variables to increase attention on under-attended tokens during generation [Chefer et al., 2023]. These approaches highlight that compositional failures are often mediated by attention and token grounding, which motivates our use of *structured supervision* (scene graphs) during training to teach more reliable binding and coverage, while keeping inference identical to standard text-to-image usage.

3.3 Spatial Conditioning with Depth

Explicit spatial conditioning provides another route to reduce relational errors by injecting geometric or layout signals into diffusion models. ControlNet introduces a parallel branch to process control signals (e.g., depth/edges/segmentation) and inject them into the original U-Net via zero-initialized layers, improving geometric consistency while preserving pretrained knowledge [Zhang et al., 2023]. T2I-Adapter achieves similar controllability with smaller adapters injected into selected U-Net layers, substantially reducing trainable parameters and making spatial control more budget-friendly [Mou et al., 2023]. These methods directly motivate our *depth-conditioned* design: we use depth as a compact spatial prior to stabilize layout and relations, but avoid adding heavy auxiliary networks by fine-tuning with LoRA.

3.4 Scene Graph Supervision for Compositional Generation

Scene graphs provide structured representations of objects, attributes, and relationships, making them a natural supervision signal for compositional generation. Early work demonstrated that structured scene representations can improve controllable image generation [Johnson et al., 2018]. More recent diffusion-based methods show that conditioning on scene graphs can reduce hallucination and improve relational fidelity by strengthening object-level grounding [Herzig et al., 2023]. DisCo further argues that complex scenes require stronger modeling of relationships and attributes, proposing specialized mechanisms to inject object-level graph information [Wang et al., 2024]. In our project, we adopt the core insight—*structured supervision improves compositionality*—but prioritize compatibility with standard text-to-image pipelines by translating scene graphs into compositional prompts and pairing them with depth signals during training.

3.5 Parameter Efficient Fine-Tuning

Finally, LoRA provides a practical mechanism to adapt large diffusion models under limited compute by learning low-rank updates to selected weight matrices [Hu et al., 2021]. Empirical studies show that the optimal rank depends on task complexity and capacity requirements [Lialin et al., 2023], motivating our systematic exploration of LoRA ranks. Concretely, we apply LoRA to U-Net attention layers and (in some configurations) to the CLIP text encoder, enabling controlled ablations of where adaptation helps or harms compositional performance under a fixed training budget.

Positioning. In summary, inference-time composition [Liu et al., 2022, Balaji et al., 2022] and training-free attention steering [Feng et al., 2023, Chefer et al., 2023] demonstrate that compositional failures are addressable but can be costly or fragile. Spatial conditioning [Zhang et al., 2023, Mou et al., 2023] and scene-graph supervision [Johnson et al., 2018, Herzig et al., 2023, Wang et al., 2024] suggest that explicit structure (geometry + relations) improves grounding. Our contribution lies at the intersection: we combine *depth-based spatial priors* with *scene-graph-derived training signals* and *LoRA-based* fine-tuning to improve compositional generation in a *single* Stable Diffusion model with standard inference.

4 Approach

4.1 Model Overview

Our approach builds upon Stable Diffusion v2.1, specifically the `SagiPolaczek/stable-diffusion-2-1-base` model from HuggingFace, mirroring the official Stable Diffusion 2.1 release. We augment it with (i) a text-guided depth predictor and (ii) depth-conditioned denoising with parameter-efficient fine-tuning. Our goal is to improve compositional prompt following (attributes, relations, numeracy, and complex compositions) while keeping standard text-to-image inference.

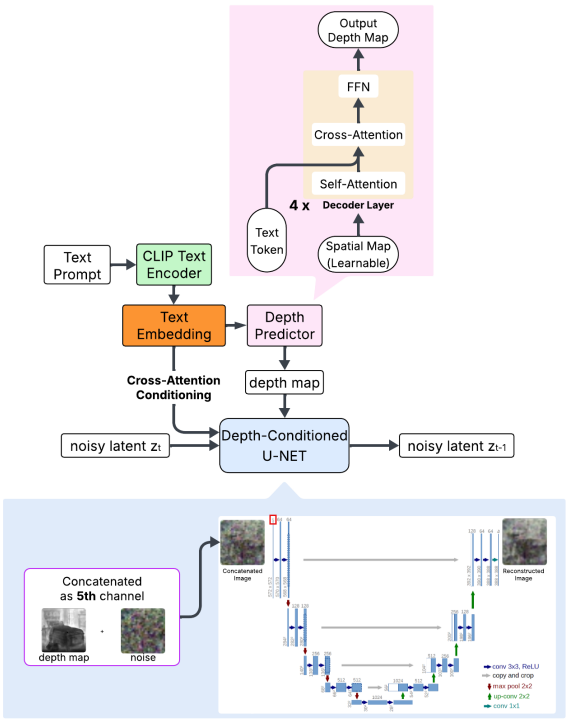

Figure 1: **One denoising step with text-to-depth conditioning.** (1) A CLIP text encoder maps the prompt to token embeddings. (2) A *text-to-depth* predictor uses learnable 64×64 spatial queries and a transformer-decoder (self-attention + cross-attention to text + FFN, repeated for 4 layers) followed by a convolutional head to produce a normalized 64×64 depth map. (3) The Stable Diffusion U-Net is modified to accept a *5-channel* input by concatenating the depth map with the 4-channel noisy latent, and predicts the noise residual to update $z_t \to z_{t-1}$. The extra input channel is initialized to preserve pretrained behavior.

4.2 Text-Guided Spatial Depth Encoding

We condition the SD2.1 U-Net on depth by extending its latent input from 4 channels to 5 channels. Concretely, we concatenate a 64×64 depth map (broadcast to latent resolution) with the noisy latent along the channel dimension and replace the first convolution (`conv_in`) to accept 5 channels, initializing the added channel to zero.

To obtain depth without requiring an external depth image at inference, we train a *text-to-depth* predictor. The predictor is a transformer-decoder that takes CLIP text token embeddings as keys/values and uses learnable spatial queries on a 64×64 grid. Its output features are projected by a lightweight convolutional head to a single-channel depth map with sigmoid normalization. For supervision, we precompute pseudo ground-truth depth maps from training images using MiDaS and downsample them to 64×64.

Training vs. inference usage. During LoRA fine-tuning of SD2.1, we condition the U-Net using the *precomputed* MiDaS depth maps from the dataset for stability. During image generation, we instead use the trained text-to-depth predictor to produce depth maps from the prompt, enabling *text-only* depth-conditioned sampling.

4.3 Training and Scene-Graph Prompt

We leverage LAION-SG scene-graph annotations as structured supervision by converting graphs into compositional prompts using templates that explicitly include objects, attributes, and relations. For example, a scene graph describing a *large red tree* next to a *small wooden bench* is rendered as: *"a large red tree next to a small wooden bench."* This increases the frequency of multi-object, attribute-rich, relation-heavy prompts during training.

We fine-tune Stable Diffusion using the standard diffusion denoising objective (MSE between predicted and target noise/velocity), with depth provided as an additional conditioning channel. We study three parameter-efficient configurations: (i) frozen text encoder + LoRA on U-Net attention, (ii) fully fine-tuned text encoder + LoRA on U-Net, and (iii) LoRA on text-encoder FFN + LoRA on U-Net attention.

4.4 Evaluation

We evaluate on the T2I-CompBench validation prompts (**2400 prompts**: 900 attribute binding, 1200 object relations, 300 complex), generating 2 images per prompt. For automatic scoring, we report CLIP similarity (OpenAI CLIP ViT-B/32) and summarize results by category (Attr/Rel/Complex). We additionally evaluate numeracy on a 300-prompt numeracy subset using a BLIP-VQA based metric that checks pairwise counting correctness. All results are reported consistently across baselines, configurations, and LoRA ranks.

5 Experiments

Research questions and hypotheses. Our experiments are designed to answer two questions: (Q1) *Where should we adapt Stable Diffusion to improve compositional prompt following under limited compute?* In particular, we test whether adapting only the U-Net (while keeping the CLIP text encoder frozen) is sufficient, or whether modifying the text encoder helps. (Q2) *How much LoRA capacity is needed?* We sweep LoRA ranks to study the trade-off between parameter efficiency and compositional performance. Our hypothesis is that (H1) freezing the CLIP text encoder preserves its pretrained semantic space and yields the most reliable gains on attribute binding and relations, while (H2) text-encoder adaptation can overfit on 100k LAION-SG pairs and harm general compositionality, but may improve numeracy by increasing sensitivity to number words.

Data, training, and hardware. We use a 9:1 train/validation split from LAION-SG with 100,000 image–text pairs (90k train / 10k val). All models are trained on an AWS `g5.xlarge` instance (single NVIDIA A10G, 24 GB). We use mixed precision and gradient checkpointing, AdamW with learning rate 1×10^{-4} and cosine annealing. For each configuration and rank, we train for one epoch on the 90k subset (runtime varies by configuration/rank; see compute section).

Evaluation protocol. We evaluate on T2I-CompBench using its standard detection-based metrics: BLIP-VQA for attribute-related questions and UniDet for relationship detection. We report four metrics: Attribute Binding, Object Relationships, Generative Numeracy, and Complex Composition (all higher is better). Our baseline is the unmodified SD v2.1 model (same sampler settings as our models), which isolates the effect of depth conditioning and LoRA adaptation.

5.1 Fine-tuning configurations

We compare three parameter-efficient fine-tuning configurations, all trained with Dual-Stream Depth Encoding and scene-graph-derived prompts:

1. **Config 1:** Frozen CLIP text encoder; LoRA on U-Net attention layers; trained text-to-depth predictor.
2. **Config 2:** Fully fine-tuned CLIP text encoder; LoRA on U-Net attention layers; trained text-to-depth predictor.
3. **Config 3:** LoRA on CLIP text encoder FFN layers *and* U-Net attention layers; trained text-to-depth predictor.

Table 1 summarizes the best-performing rank for each configuration. Config 1 provides small but consistent gains over the SD v2.1 baseline across all four metrics, supporting (H1) that depth conditioning plus U-Net LoRA improves compositionality without disrupting CLIP semantics. Config 2 underperforms Config 1 across metrics, consistent with (H2) that fully adapting the text encoder on only 100k pairs can overfit and partially degrade the pretrained semantic space. Config 3 shows a clear trade-off: it slightly lags Config 1 on attribute binding, relations, and complex prompts, but yields the strongest numeracy performance, improving numeracy from 0.2375 (baseline) to 0.5114.

Config	Attr. Bind.	Obj. Rel.	Num.	Complex
1	0.3185	0.3128	0.2713	0.3094
2	0.2770	0.2695	0.1754	0.2841
3	0.2723	0.2691	0.5114	0.2805

Table 1: Best results under each fine-tuning configuration on T2I-CompBench. All configurations use Dual-Stream Depth Encoding and scene-graph-based training. Scores range from 0 to 1 (higher is better).

5.2 LoRA Rank Sweep

To understand capacity vs. efficiency, we sweep LoRA rank $r \in \{16, 32, 64, 128\}$ under each configuration (Table 2). Across Config 1/2, higher rank does not monotonically improve performance, suggesting diminishing returns (and possible overfitting) on the 100k subset. In Config 3, numeracy varies substantially with rank and peaks at $r{=}64$, indicating that additional text-encoder capacity can disproportionately benefit counting, even when it does not help other compositional metrics.

Config	Rank	Attr. Bind.	Obj. Rel.	Num.	Complex
1	16	0.3185	0.3128	0.2713	0.3094
	32	0.3186	0.3135	0.2473	0.3090
	64	0.3181	0.3147	0.2524	0.3088
	128	0.3163	0.3129	0.2294	0.3074
2	16	0.2817	0.2700	0.1646	0.2875
	32	0.2781	0.2683	0.1700	0.2859
	64	0.2768	0.2690	0.1606	0.2841
	128	0.2770	0.2695	0.1754	0.2841
3	16	0.2765	0.2708	0.4716	0.2831
	32	0.2742	0.2677	0.4640	0.2818
	64	0.2723	0.2691	0.5114	0.2805
	128	0.2750	0.2681	0.4645	0.2786

Table 2: Rank sweep results under each configuration. Config 1 freezes the text encoder; Config 2 fully fine-tunes it; Config 3 applies LoRA to text-encoder FFN layers. All use Dual-Stream Depth Encoding and scene-graph-based training.

5.3 Baseline Comparison and Qualitative Results

Table 3 compares our best overall model (Config 1, $r{=}16$) to the SD v2.1 baseline. While gains are mod-

est, they are consistent across metrics, suggesting that the added depth channel and lightweight U-Net adaptation can improve compositional faithfulness without large-scale retraining.

Model	Attr. Bind.	Obj. Rel.	Num.	Complex
SD v2.1 Baseline	0.3173	0.3110	0.2375	0.3082
Dual-Stream Depth + LoRA (best)	0.3185	0.3128	0.2713	0.3094

Table 3: Comparison of the SD v2.1 baseline vs. our best-performing model on T2I-CompBench (higher is better).

Figure A2 provides qualitative comparisons across the baseline and our three configurations. The baseline often captures only part of the prompt (e.g., weaker attribute binding or incorrect counts). Config 1 typically produces the most faithful compositions (e.g., stronger attribute binding and more consistent layouts), while Config 2/3 remain visually plausible but more frequently exhibit attribute or relation errors, aligning with the quantitative trends in Table 1.

Limitations. These results are data- and compute-limited: we train on 100k LAION-SG pairs (far below the full dataset) and run all experiments on a single A10G GPU, which restricts the number of seeds and longer training schedules we can afford. In addition, detection-based evaluation can miss fine-grained compositional errors, so qualitative inspection remains important when interpreting small metric differences.

6 Code Overview

We implemented an end-to-end pipeline for depth-aware LoRA training, inference, and evaluation on T2I-CompBench.

- **Depth preprocessing (MiDaS) and robust data handling.** In `midas_depthmapgeneration.py`, we (i) implement fault-tolerant LAION-SG downloading (sharding, skip/log failures, resumable runs) and (ii) generate and save 64×64 MiDaS depth maps as an offline preprocessing stage (lines 17–93, 45–91).

- **Text-to-depth predictor (Transformer decoder).** In `text2depth_transformer_decoder.py`, we define a transformer-decoder predictor that maps frozen CLIP text embeddings to a dense 64×64 depth channel used for conditioning diffusion (lines 11–61).

- **Depth-aware Stable Diffusion modification and LoRA injection.** In `train_depth_lora_config1.py`, we expand the SD2.1 U-Net input from 4 to 5 channels (latent + depth) and zero-initialize the added weights to preserve pretrained behavior. We then inject LoRA into U-Net attention projections (`to_q/to_k/to_v/to_out`) and train `conv_in`; the training loop concatenates the depth channel before denoising (lines 1–31, 49–82).

- **Config-specific training regimes.** In `train_depth_lora_config2.py`, we fully fine-tune the CLIP text encoder in addition to U-Net LoRA (text encoder unfreezing: lines 55–59; optimizer: lines 40–44). In `train_depth_lora_config3.py`, we keep the base text encoder frozen but add LoRA to the text-encoder FFN layers (`fc1/fc2`), optimizing both U-Net LoRA and text LoRA (lines 55–58, 88–105).

- **Inference and reproducibility metadata.** In `generate_images_lora_text2depth_textencoder.py`, we load a configuration checkpoint, predict depth from text, concatenate depth with latents during sampling, and generate images from a prompt CSV. We also save metadata (prompt id/category/seed) for reproducibility (lines 23–47).

- **Evaluation utilities.** We provide BLIP-VQA numeracy evaluation in `eval_numeracy_blip.py` and aggregate CLIP similarities by category in `summarize_clip_by_category.py` to produce the final tables.

7 Timeline

Table 4 shows an overview of the approximate time spent on various project stages. The largest amount of time (96 hours) was spent training the model for all experiments. Preparing the dataset took 10 hours and was particularly time-consuming due to the processes of preparing depth map with MiDaS. Evaluation took 2036 hours, as it required first generating images for all test prompts across multiple configurations, followed by computing metrics (CLIP similarity, numeracy accuracy) on the large set of generated images.

Project Stage	Hours Spent
Background Literature Review	6
Understanding code from baseline methods	7
Understanding code from Stable Diffusion V2.1r implementation	4
Compiling/running existing code for baseline methods	8
Preparing dataset for our method	15
Modifying existing code and implementing our method	10
Writing scripts to run experiments	46
Training the model for all experiments	96
Evaluation for all experiments	36
Writing this document	8

Table 4: Time Spent on Various Project Stages

8 Research Log

8.1 Data Preprocessing on LAION-SG

Preprocessing LAION-SG was more difficult than expected. The HuggingFace release provides JSON entries with HTTP URLs (not packaged images), and a non-trivial portion failed due to 404s, SSL issues, or truncated downloads. To make the pipeline reliable, we rewrote the downloader to process 10k-sized shards, skip and log bad URLs, and export per-shard CSV indices so runs could be resumed after failures. Depth generation was also a bottleneck: Colab runs frequently hit timeouts and quota limits, so we migrated preprocessing to an AWS A10G instance and generated MiDaS depth maps for $\sim$100k images as a one-time offline stage.

8.2 Depth Predictors (Plan vs. Execution)

Our original plan was to benchmark several depth encoders and keep the best one. We implemented both a single-scale and a multiscale text-to-depth predictor (transformer decoder + convolutional head producing 64×64 depth). In practice, the multiscale model was heavier and showed training instability when coupled with diffusion fine-tuning under our compute budget. To ensure stable full sweeps across settings, we finalized on the single-scale predictor and used the multiscale variant only for small pilots. This reduced architectural comparison breadth, but improved reproducibility and throughput.

8.3 LoRA Sweeps and Findings

We integrated depth into SD2.1 by expanding the U-Net input from 4 to 5 channels and zero-initializing the added weights to preserve pretrained behavior. We then evaluated three regimes—**Config 1** (frozen text encoder + LoRA on U-Net attention), **Config 2** (full text-encoder fine-tuning + LoRA on U-Net), and **Config 3** (LoRA on text-encoder FFN + LoRA on U-Net)—sweeping ranks $\{16, 32, 64, 128\}$. Running all 3×4 settings with T2I-CompBench plus numeracy was more time-consuming than planned, and we had to switch from T4 to A10G due to memory overflows. Consequently, we deprioritized some intended ablations (e.g., depth-free LoRA and multi-seed variance) to complete one consistent, end-to-end evaluation pipeline (generation $\rightarrow$ CLIP scoring by category $\rightarrow$ BLIP-VQA numeracy).

The results revealed a clear pattern. **Config 1** produced small but consistent gains over the SD2.1 baseline across all four metrics, suggesting that depth conditioning helps compositional reasoning when the pretrained CLIP semantic space remains intact. **Config 2** and **Config 3** did not beat Config 1 on attribute binding, relations, or complex prompts; we hypothesize that adapting CLIP on only $\sim$100k LAION-SG pairs can overfit and distort the original embedding geometry. The main exception was **numeracy**: **Config 3** achieved the best counting performance, indicating a trade-off where text-encoder LoRA increases sensitivity to number words but can slightly weaken binding/relations. Rank effects were not monotonic, consistent with capacity interacting with optimization stability under limited data/compute.

9 Conclusion and Future Work

Please see Appendix A.1 for details.

10 Thought-Experiment on Compute

10.1 Actual Compute Usage

All experiments were run on **AWS g5.xlarge** ($1\times$NVIDIA **A10G**, 24 GB). We approximate the on-demand cost as **$1.0 per A10G GPU hour**. Total usage:

- **Baseline evaluation**: ~ 8 A10G GPU hours.
- **Depth preprocessing** (MiDaS for $\sim$100k images): ~ 16 A10G GPU hours.
- **Training** (3 configs $\times$ 4 ranks, ~ 8 hours each): $3 \times 4 \times 8 = 96$ A10G GPU hours.
- **Evaluation** (3 configs $\times$ 4 ranks, ~ 3 hours each): $3 \times 4 \times 3 = 36$ A10G GPU hours.

Total: $8 + 16 + 96 + 36 = 156$ A10G GPU hours $\Rightarrow$ **$** $155 - 160$ estimated compute cost.

10.2 Hypothetical Additional $450 Budget

At the same rate, **$450** ≈ 450 **A10G GPU hours**. We would spend it to strengthen conclusions (scale, ablations, and reliability):

1. **Scale best setting** ($\sim$ 200 *A10G GPU hours*): train Config 1 on a larger LAION-SG subset (200k–500k) and longer schedules to test whether gains persist with more data/epochs.
2. **Ablations + variance** (~ 150 *A10G GPU hours*): depth-free LoRA baselines (remove depth channel/predictor) and multi-seed reruns to report variance/confidence for CLIP + numeracy metrics.
3. **Stronger evaluation** ($\sim$ 100 *A10G GPU hours*): more samples per prompt / more steps (and curated qualitative grids) to reduce metric noise and better diagnose failure modes.

References

Yogesh Balaji, Seungjun Nah, Xun Huang, Arash Vahdat, Jiaming Song, Qinsheng Zhang, Karsten Kreis, Miika Aittala, Timo Aila, Samuli Laine, Bryan Catanzaro, Tero Karras, and Ming-Yu Liu. ediff-i: Text-to-image diffusion models with an ensemble of expert denoisers. *arXiv preprint arXiv:2211.01324*, 2022.

Hila Chefer, Yuval Alaluf, Yael Vinker, Lior Wolf, and Daniel Cohen-Or. Attend-and-excite: Attention-based semantic guidance for text-to-image diffusion models. *ACM Transactions on Graphics*, 42(4):148, 2023.

Weixi Feng, Xuehai He, Tsu-Jui Fu, Varun Jampani, Arjun Akula, Pradyumna Narayana, Sugato Basu, Xin Eric Wang, and William Yang Wang. Training-free structured diffusion guidance for compositional text-to-image synthesis. *arXiv preprint arXiv:2212.05032*, 2023.

Omri Herzig, Omer Bar-Tal, Or Patashnik, Dani Lischinski, Daniel Cohen-Or, and Shai Avidan. Spatext: Spatio-textual representation for controllable image generation. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

Edward J Hu, Shen Yelong, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Kaiyi Huang, Kaiyue Sun, Enze Xie, Zhenguo Li, and Xihui Liu. T2i-compbench: A comprehensive benchmark for open-world compositional text-to-image generation. *arXiv preprint arXiv:2307.06350*, 2023.

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*, 2018.

Zejian Li, Chenye Meng, Yize Li, Ling Yang, Shengyuan Zhang, Jiarui Ma, Jiayi Li, Guang Yang, Changyuan Yang, Zhiyuan Yang, et al. Laion-sg: An enhanced large-scale dataset for training complex image-text models with structural annotations. *arXiv preprint arXiv:2412.08580*, 2024.

Vladislav Lialin, Vijeta Deshpande, and Anna Rumshisky. Scaling down to scale up: A guide to parameter-efficient fine-tuning. *arXiv preprint arXiv:2303.15647*, 2023.

Nan Liu, Shuang Li, Yilun Du, Antonio Torralba, and Joshua B Tenenbaum. Compositional visual generation with composable diffusion models. *arXiv preprint arXiv:2206.01714*, 2022.

Chong Mou, Xintao Wang, Liangbin Xie, Jian Zhang, Zhongang Qi, Ying Shan, and Xiaohu Qie. T2i-adapter: Learning adapters to dig out more controllable ability for text-to-image diffusion models. *arXiv preprint arXiv:2302.08453*, 2023.

Yunnan Wang, Ziqiang Li, Zequn Zhang, Wenyao Zhang, Baao Xie, Xihui Liu, Wenjun Zeng, and Xin Jin. Scene graph disentanglement and composition for generalizable complex image generation. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. *arXiv preprint arXiv:2302.05543*, 2023.

A Appendix

Figure A1: Example of compositional failure. Generated using Stable Diffusion with prompt: "The sharp blue scissors cut through the thick white paper."

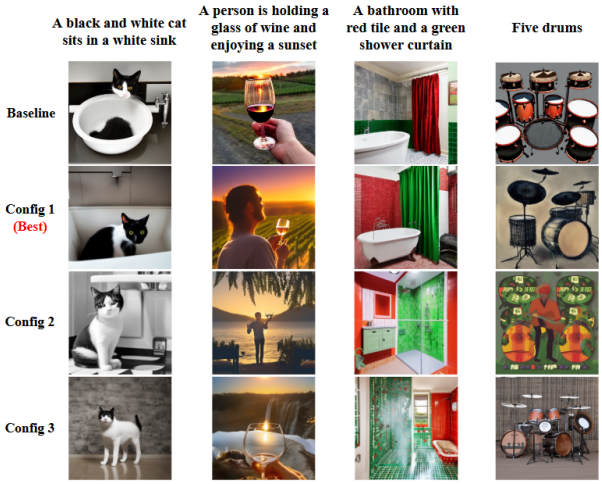

Figure A2: Qualitative comparison of the SD 2.1 baseline and our three fine-tuning configurations on four compositional prompts (columns). Each row shows images from the baseline model and Config 1–3 (top to bottom). Config 1 most consistently preserves object attributes, relations, and counts, while Config 2 and Config 3 often lose parts of the specified composition.

Qualitative analysis of configurations. Figure A2 illustrates the differences between the baseline and our three fine-tuning configurations on four representative prompts. For *"A black and white cat sits in a white sink"*, the baseline partially mixes the colors of the cat and the sink, and Config 2/3 essentially ignore the white sink and focus only on the cat. Only Config 1 preserves both the black-and-white fur and a clearly white sink. For *"A person is holding a glass of wine and enjoying a sunset"*, the key relation is the *holding* action: the baseline and Config 3 rarely show a realistic hand gripping the glass, and Config 2 sometimes produces a floating glass, whereas Config 1 reliably depicts a person actually holding the glass. For *"A bathroom with red tile and a green shower curtain"*, Config 2 and Config 3 smear red and green across multiple surfaces, and the baseline tends to swap the colors of tiles and curtain. Only Config 1 simultaneously produces red tiles and a green shower curtain as specified. Finally, for *"Five drums"*, the baseline and Config 2 often generate the wrong number of drums, while Config 1 and Config 3 more frequently match the requested count, in line with the numeracy metrics. Overall, these qualitative results are consistent with our quantitative scores: under our current setup, Config 1 provides the best trade-off across attribute binding, relational reasoning, and numeracy, and we therefore adopt it as our final configuration.

A.1 Conclusion and Future Work

We studied whether text-guided depth conditioning plus parameter-efficient LoRA can improve compositional generation in Stable Diffusion 2.1. Across three fine-tuning configurations and LoRA ranks $\{16, 32, 64, 128\}$, **Config 1** (frozen CLIP text encoder + LoRA on U-Net attention + depth channel) was the most reliable, delivering *small but consistent* gains over the baseline across attribute binding, object relations, numeracy, and complex prompts, suggesting depth can help composition when CLIP semantics are preserved. In contrast, adapting the CLIP text encoder (**Config 2** full fine-tuning; **Config 3** text-encoder LoRA) did not improve the CLIP-based compositional metrics, consistent with overfitting or semantic drift when tuning CLIP on only $\sim$100k LAION-SG pairs; the main exception is **numeracy**, where **Config 3** performs best, indicating a trade-off where text-encoder adaptation helps counting but can hurt binding and relations. Limitations include the 100k data subset (vs. full LAION-SG), a smooth 64×64 text-to-depth predictor, and a single A10G GPU budget that prevented depth-free baselines and multi-seed variance estimates; future work should scale data/epochs for Config 1 while keeping CLIP frozen, explore higher-fidelity/multiscale depth conditioning, add depth-free and multi-seed ablations, and complement CLIP/BLIP metrics with targeted qualitative or MLLM-based assessment of compositional failures.