LoRAtorio: An intrinsic approach to LoRA Skill Composition

Niki Foteinopoulou[1*] **Ignas Budvytis**[2] **Stephan Liwicki**[1*]
[1]Cambridge Research Laboratory, Toshiba Europe
[2]Independent Researcher

Abstract

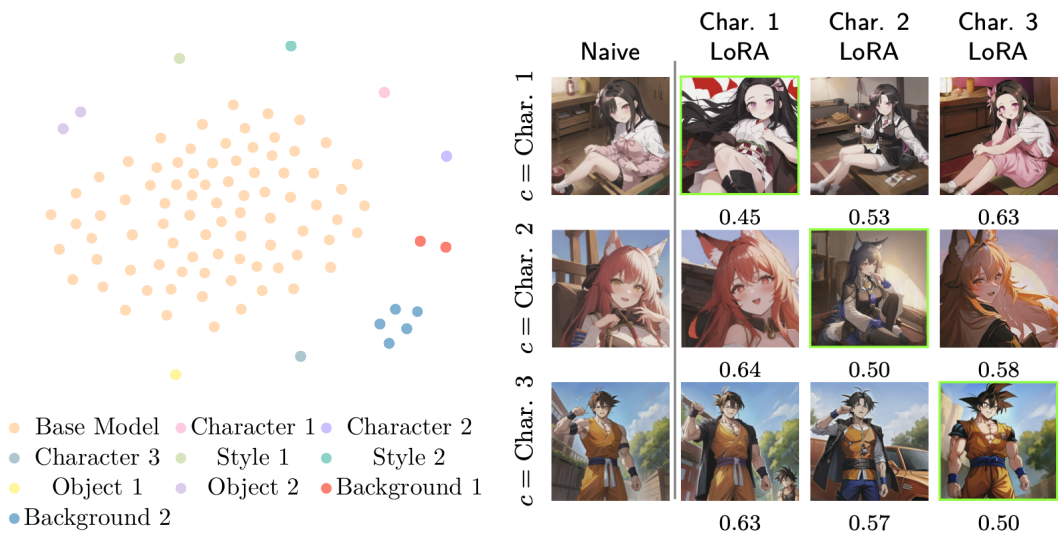
Low-Rank Adaptation (LoRA) has become a widely adopted technique in text-to-image diffusion models, enabling the personalisation of visual concepts such as characters, styles, and objects. However, existing approaches struggle to effectively compose multiple LoRA adapters, particularly in open-ended settings where the number and nature of required skills are not known in advance. In this work, we present LoRAtorio, a novel train-free framework for multi-LoRA composition that gates each adapter using only signals already inside the denoiser, specifically, patch-level cosine (dis)similarity between each LoRA's predicted noise and the base model's. We refer to this as "intrinsic" guidance because no external parameters are introduced. Our method is motivated by two key observations: (1) LoRA adapters trained on narrow domains produce unconditioned denoised outputs that diverge from the base model, and (2) when conditioned out of distribution, LoRA outputs show behaviour closer to the base model than when conditioned in distribution. These patch-level similarities are used to construct a spatially-aware weight matrix, which guides a weighted aggregation of LoRA outputs. To address domain drift, we further propose a modification to classifier-free guidance that incorporates the base model's unconditional score into the composition. We extend this formulation to a dynamic module selection setting, enabling inference-time selection of relevant LoRA adapters from a large pool. On the *ComposLoRA* benchmark, LoRAtorio achieves state-of-the-art performance, with CLIPScore that does not deteriorate as the number of composed adapters grows, surpassing the strongest prior method by up to 1.3% in CLIPScore, and reaching a 76.92% GPT-4V win rate against the closest baseline. We further show that the framework is backbone-agnostic: the intrinsic-similarity principle transfers to rectified-flow architectures (FLUX.1-dev) without retraining. Project page: `https://nickyfot.github.io/loratorio-skill-composer/`.

1 Introduction

Diffusion models operate by gradually learning to reverse a noise process, effectively capturing the underlying data distribution through iterative denoising [10, 30, 42]. In practice, this enables them to approximate the complex structure of their training data and generate new, previously unseen samples that remain faithful to the original data's domain. Beyond base text-to-image generation capabilities, works such as Dreambooth [35] and StyleDrop [41] have enabled personalisation and fine-grained customisation. These approaches often rely on LoRA adapters [11], which specialise a base model to preserve the identity of specific concepts

*Corresponding authors: `niki.foteinopoulou@toshiba.eu`, `stephan.liwicki@toshiba.eu`

40th Conference on Neural Information Processing Systems (NeurIPS 2026).



(a) (b)

Figure 1: Observations of noise prediction from the *ComposLoRA* testbed. (a) t-SNE visualisation of the unconditioned latent space, for the Base Model and LoRA-adapted models. (b) Generated images from each Character LoRA, conditioned with text prompts originally associated with other LoRAs. When a prompt falls outside a LoRA's training distribution, the predicted latent $e_{\theta_i}(z_0, 0, c)$ of the ith LoRA tends to align closely with that of the base (Naïve) model, showing minimal deviation due to changes in $p_{\theta_i}(x)$, also shown by the cosine similarity of the conditioned noise prediction of the Naïve model with that of the ith LoRA.

or objects, supporting applications like virtual try-on [27] and avatar generation [14]. Each LoRA adapter effectively encodes a "skill" or concept, and generation with a single adapter yields precise, high-quality outputs. However, when multiple skills are loaded simultaneously into a single model instance, we observe a rapid deterioration in performance [46, 32]. Understanding the source of this degradation is key to enabling reliable multi-concept generation.

To better understand the challenges of composing multiple LoRA adapters, we begin with a preliminary analysis of their behaviour. Specifically, we examine the unconditional noise representations produced by the base model and various LoRA-augmented models. We observe that the distribution of the LoRA diverges from that of the base model (Figure 1a), particularly when LoRAs are trained on narrow or highly specialised datasets, conditions common in personalisation settings [23, 35]. This domain shift also manifests in conditioned outputs, as evident through visual inspection in Figure 1b (details in Appendix C).

Observation 1: The unconditioned noise estimate $e_i(z, t)$ produced by the ith LoRA differs from that of the base model $e(z, t)$.

While this divergence is notable, especially under unconditional or in-domain conditions, LoRA adapters are also known to mitigate catastrophic forgetting, particularly compared to fully fine-tuned models [1]. That is, LoRAs tend to preserve the base model's generalisation capabilities. Indeed, we observe that even though there are stylistic changes in the generated image as a result of the loaded LoRAs, the composition and theme of the generated output more closely resemble the base model when a text condition outside the LoRA distribution is given. This is attributed to the sparse and low-norm nature of LoRA weights [6, 37]. To quantify this effect, we measure cosine similarities between the noise scores of LoRA-augmented models and the base model, both within and outside LoRA's training distribution. These measurements show high similarity to the base model out-of-distribution and lower similarity in-distribution, which, together with the visual inspection in Figure 1b, support the following:

Observation 2: When the input condition lies outside the LoRA's target domain, the output of the augmented model more closely resembles that of the base model.

Previous skill composition research utilises **trainable** methods, such as mixture-of-experts or hyper-networks [2, 38, 47], and **train-free** approaches involving scheduling, latent merging, or weight arithmetic [46, 48, 21, 44, 15, 12, 37]. Trainable methods often prove impractical due to the high cost of retraining for new concepts and data confidentiality constraints. Conversely, existing train-free techniques, relying on manual schedules or unweighted score aggregation, suffer from performance degradation as the number of skills increases [32, 46]. Critically, all prior works assume the required LoRAs are pre-specified. This is unfeasible in real-world settings like personalised advertising, where user intent is only known at inference time, and the combinatorial space of skill mixtures is vast.

Motivated by Observations 1 and 2, we propose LoRAtorio, a train-free method that gates LoRA contributions using their patch-level divergence from the base model in latent space. Per-patch divergence is converted via a SoftMin into spatially-aware gating weights, so each LoRA contributes most where it differs most from the base; a re-centered classifier-free guidance further corrects for unconditional drift by mixing in the base model's unconditional score. Unlike prior approaches that rely on extrinsic signals such as RGB frequency [48] or hand-crafted scheduling [46], LoRAtorio is entirely intrinsic; the gating signal is the consistency between LoRA and base representations themselves. We further extend the task to a *dynamic* module-selection setting, where all available adapters are loaded, and the relevant subset is chosen at inference time, reflecting real-world scenarios in which the required skills are not known a priori.

Our main contributions can be summarised as follows:

- We introduce **LoRAtorio**, a train-free method that aggregates multiple LoRA adapters using a patch-level dissimilarity score between each LoRA's predicted noise and the base model's, the first composition mechanism gated by an intrinsic, training-free metric.
- Furthermore, we propose a modification to classifier-free guidance that re-centers the unconditional score on the base model (Section 2.2, Figure 3), correcting the domain drift implied by Observation 1.
- We further introduce a dynamic module-selection regime in which a candidate pool of LoRAs is loaded simultaneously and the relevant subset is chosen at each denoising step from the same intrinsic similarity signal, removing the need for prompt-time scheduling.

We demonstrate that LoRAtorio achieves state-of-the-art (SoTA) performance on the *ComposLoRA* benchmark both in terms of automated metrics and human preference. This is consistent for both static and dynamic module settings. As a further extension demonstrating the generality of the intrinsic-similarity principle, we adapt the framework to a rectified-flow [4] backbone (FLUX.1-dev), the first spatially-aware multi-LoRA composition method on rectified-flow backbones.

2 LoRAtorio

Our skill composition (Section 2.1) is motivated by Observation 2 (Figure 1b), while the re-centering guidance (Section 2.2) addresses Observation 1 (Figure 1a).

Preliminaries: Latent Diffusion Models [33] operate by performing the denoising diffusion process in a learned latent space. Given an input x_0, an encoder $\mathcal{E}$ maps it to a latent representation $z_0 = \mathcal{E}(x_0)$; during the diffusion process, Gaussian noise is progressively added to z_0, thus producing a noisy sequence $\{z_t\}_{t=1}^{T}$. The diffusion model learns to approximate the reverse process via a denoising network $e_\theta(z_t, t, c)$, conditioned on context c. Classifier-free guidance (CFG) [9] is incorporated by training the model with both conditional and unconditional objectives; guidance is applied by modifying the predicted noise as follows:

$$\hat{e}_\theta(z_t, t, c) = e_\theta(z_t, t) + s \cdot (e_\theta(z_t, t, c) - e_\theta(z_t, t)), \tag{1}$$

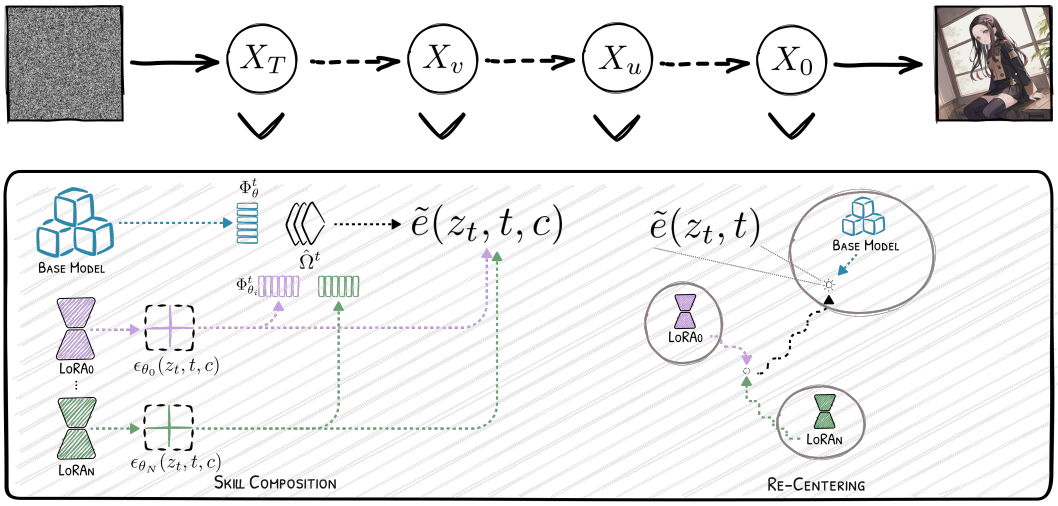


Figure 2: Overview of LoRAtorio. **Skill Composition:** At each denoising timestep t, the conditional score from the ith LoRA, $e_{\theta_i}(z_t, t, c)$, is partitioned into P spatial patches (equation 3). Each patch is flattened and compared to its corresponding patch in the base model's predicted noise using cosine similarity. The resulting similarity matrix is passed through a SoftMin function (equation 4) to produce a weight matrix Ω, assigning higher weights to patches that diverge more from the base model. These weights are used to compute a spatially-aware weighted average across LoRA outputs (equation 5 and equation 6). **Re-Centering:** To mitigate domain drift of the unconditional score by the multiple LoRAs, we alter classifier-free guidance by incorporating a weighted combination of the base model's unconditional score and the aggregated LoRA score (equation 7).

where $s \geq 1$ is the guidance scale and $e_\theta(z_t, t)$ denotes the unconditional prediction. This approach allows the model to maintain sample diversity while enhancing conditional fidelity without relying on external classifiers.

Low-Rank Adaptation (LoRA) [11] is a parameter-efficient fine-tuning method that enables adaptation of large models by injecting trainable low-rank matrices into existing weight layers. In the context of diffusion models, LoRA allows modification of the model's behaviour (e.g., emphasising identity features) without altering the full parameter set, thereby reducing the risk of catastrophic forgetting. Specifically, for a weight matrix $W \in \mathbb{R}^{d\times k}$, LoRA introduces a trainable update $\Delta W = AB$, where $A \in \mathbb{R}^{d\times r}$ and $B \in \mathbb{R}^{r\times k}$, with $r \ll \min(d, k)$. This decomposition allows the model to adapt key features—such as identity attributes—by updating only a small number of parameters; however, when multiple LoRA adapters are present, a linear combination of the weights may lead to semantic conflicts and reduced image quality [12, 46, 48]. To address issues related to weight manipulation techniques, [46] proposes aggregating conditional and unconditional scores using a weighted average, so that for N LoRAs:

$$\tilde{e}(z_t, t, c) = \frac{1}{N}\sum_{i=0}^{N} w_i \cdot [e_{\theta_i}(z_t, t) + s \cdot (e_{\theta_i}(z_t, t, c) - e_{\theta_i}(z_t, t))] \tag{2}$$

where the weights w are a scalar hyperparameter (set to 1).

2.1 Skill Composition using intrinsic knowledge

We propose **LoRAtorio**, a method that activates all LoRAs at each timestep by leveraging the similarity between their noise latent representations and that of the base model. Motivated by Observation 2, we compute the cosine similarity between the output of the model after incorporating the ith LoRA, denoted by $e_{\theta_i}(z_t, t, c)$, and the base model's output $e_\theta(z_t, t, c)$.

Since the conditioned predicted noise $e(z_t, t, c)$ retains spatial structure, we first perform channel-wise averaging to reduce the dimensionality from $H \times W \times C$ to $H \times W$. We then

partition each of these 2D maps into $P = (H \cdot W)/d^2$ non-overlapping patches of size $d \times d$ and flatten each patch into a vector of length d^2. Let $\phi(\cdot) : \mathbb{R}^{H \times W} \to \mathbb{R}^{P \times d^2}$ denote this process. We denote the resulting tokenised outputs as:

$$\Phi_\theta^t = \phi(e_\theta(z_t, t, c)), \quad \Phi_{\theta_i}^t = \phi(e_{\theta_i}(z_t, t, c)) \tag{3}$$

with $\Phi_\theta^t, \Phi_{\theta_i}^t \in \mathbb{R}^{P \times d^2}$. For each LoRA i, we compute the cosine similarity between the corresponding patch vectors of Φ_θ^t and $\Phi_{\theta_i}^t$, resulting in a weight matrix $\Omega^t = \left\langle \Phi_\theta^t, \Phi_{\theta_i}^t \right\rangle_{\cos} \in \mathbb{R}^{N \times P}$ where N is the number of LoRAs. We then apply a SoftMin operation along the N dimension:

$$\hat{\Omega}^t = \text{softmin}_\tau(\Omega^t), \quad \text{where} \quad \text{softmin}_\tau(x) = \frac{\exp(-x_i/\tau)}{\sum_{j=1}^N \exp(-x_j/\tau)} \tag{4}$$

and $\tau > 0$ is the temperature parameter controlling the softness of the SoftMin. This makes the weighting interpretable as a soft attention mechanism, where LoRAs that diverge more from the base model are given stronger influence in regions where they are more confident. Note the use of SoftMin: low cosine similarity, i.e. greater divergence from the base model, yields a small input value and therefore a high gating weight, in line with our interpretation of divergence as evidence of in-distribution adapter activity. We upscale $\hat{\Omega}^t \in \mathbb{R}^{N \times (H/d \cdot W/d)}$ to match the spatial resolution of the original feature map using a Kronecker product:

$$\hat{\Omega}^{t,\text{up}} = \hat{\Omega}^t \otimes \mathbf{1}_{d \times d} \tag{5}$$

where $\mathbf{1}_{d \times d}$ is a matrix of ones. This operation effectively repeats each similarity value over a $d \times d$ block. The upscaled similarity maps are then used to modulate the expert outputs during denoising. The final conditional estimate is computed as a weighted average:

$$\tilde{e}(z_t, t, c) = \sum_{i=1}^N \hat{\Omega}_i^{t,\text{up}} e_{\theta_i}(z_t, t, c) \tag{6}$$

We interpret cosine similarity in the noise prediction in the latent space as a proxy for LoRA confidence or relevance: patches where LoRA strongly deviates from the base model are assumed to reflect greater domain-specific influence. Because diffusion features strongly correlate with spatial layout, localised divergence indicates active, in-domain modification by the adapter. This follows from the low-rank structure of LoRA, as the update $\Delta W = AB$ has a negligible effect on inputs lying outside its learned subspace, which often correspond to out-of-distribution inputs. As such, high similarity indicates that the adapter is inactive, whereas lower similarity suggests in-distribution behaviour. A theoretical motivation for similarity-based weighting is included in Appendix B.

2.2 Re-centering guidance

To address Observation 1, we note that each LoRA θ_i implicitly encodes its own training distribution $p_{\text{LoRA}_i}(x)$, so its unconditional noise $e_{\theta_i}(z_t, t)$ diverges from the base $e_\theta(z_t, t)$ (Figure 1a). Since CFG (equation 1) extrapolates between conditional and unconditional terms, the guidance term $e_{\theta_i}(z_t, t, c) - e_{\theta_i}(z_t, t)$ becomes a biased estimator of $\nabla_x \log p(x|c) - \nabla_x \log p(x)$, skewed by $p_{\text{LoRA}_i}(x)$. With multiple LoRAs active, these unconditional terms can additionally conflict, lowering subject fidelity under standard CFG.

To mitigate this drift, we propose "re-centering" the guidance computation by incorporating the unconditional base model output. Specifically, we use the average of the base model and LoRA-weighted unconditional outputs in CFG so that the final collective guidance $\hat{e}(z_t, t, c)$ is then calculated as follows:

$$\begin{aligned} \tilde{e}(z_t, t) &= \lambda \sum_{i=1}^N \hat{\Omega}_i^{t,\text{up}} e_{\theta_i}(z_t, t) + (1 - \lambda) e_\theta(z_t, t) \\ \hat{e}(z_t, t, c) &= \tilde{e}(z_t, t) + s\left[\tilde{e}(z_t, t, c) - \tilde{e}(z_t, t)\right]. \end{aligned} \tag{7}$$

We fix $\lambda = 0.5$ across all experiments, informed by a qualitative ablation (Appendix D.1). A visual representation of the re-centering method can be seen in Figure 3.

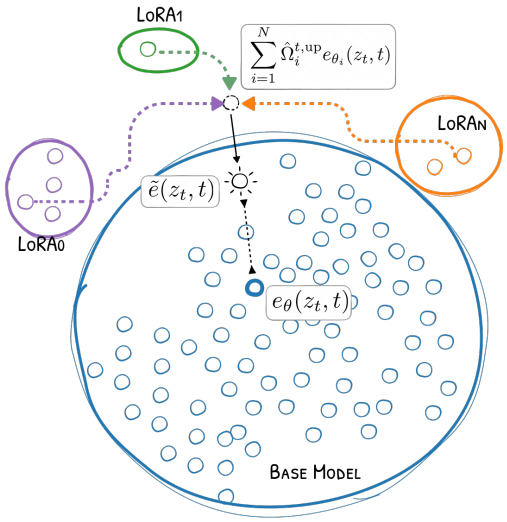


Figure 3: Visualisation of the effect of re-centering guidance (equation 7) on the unconditional noise score. Re-centering ensures the $p(x|c)$ is not over-emphasising implausible or under-trained regions of the collective data distribution after CFG. When the scores are similar, the transformation is not significant, but when there is a large adjustment, the difference is in the direction towards more probable samples.

2.3 Dynamic module selection

The Multi-LoRA composition task is defined under the assumption that only a known subset of LoRA adapters—those relevant to the current generation task—are loaded. This restricts flexibility, as it requires prior knowledge of which LoRAs are needed, and contradicts the goal of a truly inference-time, modular composition system. We propose expanding the task to a dynamic selection setting, where all available LoRA adapters are loaded into the model that dynamically selects which ones to activate based on the input. To address the dynamic setting, we propose using only the top-k most distant LoRAs at each timestep t. We first perform a hard masking step by selecting the top-k most relevant LoRA experts using a similarity-based gating metric Ω^t. Specifically, we compute:

$$\begin{aligned} \mathcal{I}_k &= \text{TopK}(1 - \Omega^t, k) \\ \tilde{\Omega}_i^t &= \begin{cases} \Omega_i^t & \text{if } i \in \mathcal{I}_k \\ \infty & \text{otherwise} \end{cases} \\ \hat{\Omega}_i^t &= \text{softmin}_\tau(\tilde{\Omega}^t) \end{aligned} \tag{8}$$

The $\hat{\Omega}_i^t$ is then upscaled and reshaped as described in Section 2.1, so that it can be used in the subsequent weighted average and re-centering steps. For fair comparison against the static setting, we set $k = C$ where C is the number of distinct concepts in the prompt; a fully zero-shot variant replacing TopK with a similarity threshold is left to future work. We include exploratory retrieval results for such a variant in Appendix I.

3 Experimental Results

3.1 Implementation Details

For our experiments, we follow the setup of [46], using *stable-diffusion-v1.5* [33] as the backbone for all *ComposLoRA* tests. We use the "Realistic_Vision_V5.1" and "Counterfeit-V2.5" checkpoints for realistic and anime-style images, respectively. For experiments with a Flux base model [20], we use the "black-forest-labs/FLUX.1-dev" checkpoint. For the realistic subset, we use 100 denoising steps, a guidance scale $s = 7$, and image size 1024×768; for the anime subset, we use 200 steps, $s = 10$, and 512×512 resolution. DPM-Solver++ [28]

Table 1: CLIPScore of LoRAtorio against previous composition methods on *ComposLoRA*.

Model	$N = 2$	$N = 3$	$N = 4$	$N = 5$	Avg.
Naïve [33]	35.014	34.927	34.384	33.809	34.534
Merge [15]	33.726	34.139	33.399	32.364	33.407
Switch [46]	35.394	35.107	34.478	33.475	34.614
Composite [46]	35.073	34.082	34.802	32.582	34.135
LoraHub [12]	35.681	35.127	34.970	33.485	34.816
Switch-A [48]	35.451	35.383	34.877	33.366	34.769
CMLoRA [48]	35.422	35.215	35.208	34.341	35.047
MultLFG [34]	**36.570**	36.125	36.180	35.920	36.199
LoRAtorio	35.236	**36.426**	**37.136**	**36.626**	**36.356**

is used as the sampler, with all LoRAs scaled by a weight of 0.8. We empirically set an adaptive temperature $\tau = 1/((T - t) * 10)$, which decreases as denoising progresses, yielding softer gating at high-noise steps and sharper attribution as spatial structure emerges. The ablation in Table 7 shows constant $\tau = 1$ shifts CLIPScore by less than 0.25 points, within seed variance, yet produces visibly inferior identity preservation at concept boundaries. For all experiments, we set the size of each patch to 2×2. Since our method operates at inference time, all experiments are run on a single RTX A6000 GPU. Results are averaged over three runs.

3.2 CLIPScore

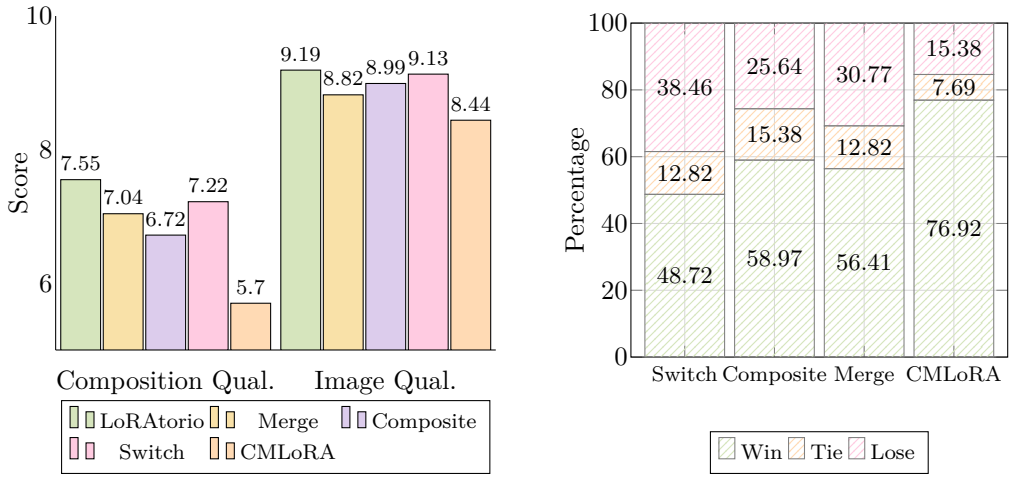
We employ CLIPScore [8] to evaluate how well the generated images match the text prompt, shown in Table 1. Even though CLIPScore does not evaluate compositional quality, acting more as a bag of words [46], it is still an important indicator of text-to-image fidelity. The multi-LoRA composition task becomes monotonically harder as N increases: each additional adapter expands the space of inter-LoRA conflicts, increases the proportion of patches that lie out-of-distribution for at least one adapter, and amplifies cumulative domain drift in the unconditional score (Figure 1a). This is reflected in the CLIPScore profile of every prior method, all of which degrade from $N = 2$ to $N = 5$. LoRAtorio is the only method whose CLIPScore does not collapse with N, surpassing the strongest prior method by up to 1.3% at $N \geq 3$ and remaining competitive at $N = 2$ where adversarial conflicts are minimal. We view robustness across N rather than absolute performance at any single N as the meaningful axis of evaluation. A breakdown by subset can be seen in Appendix D, and an ablation in Appendix D.1.

3.3 GPT-4V Evaluation

To assess the compositional and aesthetic qualities of our method, we employ a GPT-4V -based evaluation as outlined in the *ComposLoRA* testbed [46], scoring LoRAtorio against Switch, Composite, Merge and CMLoRA across two dimensions, "Composition Quality" and "Image Quality". MultLFG [34] cannot be included in qualitative comparisons as neither code nor evaluation images have been released at the time of writing. Scores and Win Rates can be seen in Figure 4a and Figure 4b, respectively. LoRAtorio outperforms previous works both in terms of average scores and win rate, i.e. pairwise comparison, closely followed by Switch. Additional results, including qualitative comparisons that corroborate this ranking, can be seen in Appendix D.

3.4 Human Evaluation

Further to the GPT-4V evaluation, we employ human experts to assess LoRAtorio qualitatively against previous works, as described by [48] across four criteria: Element Integration, Spatial Consistency, Semantic Accuracy, and Aesthetic Quality. As with the GPT-4V evaluation, MultLFG is omitted because no source code or evaluation images are publicly available. The results shown in Table 2 corroborate the GPT-4V evaluation, with LoRAtorio



(a) Composition and Image Quality comparison.

(b) Overall win rate of LoRAtorio.

Figure 4: GPT-4V Evaluation on *ComposLoRA* against previous SoTA works.

outperforming all previous works closely followed by Switch. Details of the interface and definitions for the human evaluation can be seen in Appendix K.

Table 2: Human Evaluation of our Method against previous SoTA along four qualitative axes.

	Element Integration	Spatial Consistency	Semantic Accuracy	Aesthetic Quality
LoRAtorio	**7.64**	**7.58**	**7.33**	**6.83**
CMLoRA	5.63	5.58	6.08	5.25
Composite	6.46	6.71	6.71	6.46
Switch	7.57	7.50	6.88	6.71
Merge	6.83	6.71	6.58	6.08

3.5 Dynamic Module Selection

When all LoRAs are loaded simultaneously, LoRA Merge collapses to incoherent outputs (Table 3 and Appendix J): even sparse, low-norm LoRA weights interfere meaningfully with denoising under out-of-distribution conditioning, motivating reliable per-step selection. LoRAtorio maintains high CLIPScore in this setting, with minimal influence from unrelated adapters. We provide a comparison of the computational cost (MACs and latency) against previous works in Appendix I.

Table 3: CLIPScore of LoRAtorio against previous SoTA on *ComposLoRA* in a dynamic module selection setting, where N is the number of LoRA experts needed.

	$N = 2$	$N = 3$	$N = 4$	$N = 5$	Avg.
LoRAtorio	**34.593**	**35.563**	**36.480**	**37.028**	**35.916**
Naïve	35.014	34.927	34.384	33.809	34.534
Merge	27.167	27.151	27.023	27.272	27.153

3.6 Extension to Rectified-Flow Architectures

Our method is architecturally agnostic: any latent diffusion model whose denoiser produces a structured noise prediction is compatible with patch-level similarity gating. To illustrate this generality, we extend LoRAtorio to a rectified-flow backbone [26], FLUX.1-dev. As FLUX is transformer-based and its latent space is natively tokenised, we apply the gating directly on the token outputs and omit explicit patch tokenisation; FLUX also embeds guidance signals

inside cross-attention rather than via the standard CFG construction, so the re-centering step (which targets CFG drift) is structurally inapplicable and is omitted (Appendix G). The FLUX baselines available out-of-the-box are limited, as adapting prior multi-LoRA methods to rectified-flow requires non-trivial re-engineering. The purpose of this evaluation is therefore to demonstrate the principle's portability rather than to produce a head-to-head benchmark; Table 4 confirms that the trends from SD1.5 transfer cleanly to a stronger, fundamentally different backbone.

Table 4: CLIPScore of LoRAtorio against selected composition methods, using Flux architecture.

(a) Static Modules

Model	$N = 2$	$N = 3$	$N = 4$	$N = 5$	Avg.
Naïve	33.125	34.999	37.048	38.568	35.935
Merge	33.733	35.134	35.830	36.590	35.322
LoRAtorio	**33.992**	**36.033**	**37.781**	**39.368**	**36.794**

(b) Dynamic Module Selection

	$N = 2$	$N = 3$	$N = 4$	$N = 5$	Avg.
Naïve	33.125	34.999	37.048	38.568	35.935
Merge	25.850	27.858	29.726	30.997	28.608
LoRAtorio	**33.284**	**35.753**	**37.910**	**38.861**	**36.452**

4 Related Work

4.1 Text-to-image Generation

Composable image generation aims to faithfully integrate multiple user-specified concepts. Early methods relied on layout or scene-graph conditioning [17, 42, 7], while recent works modify the diffusion generative process for multi-concept prompts [5, 13, 19, 24, 31]. However, these approaches often require prompt engineering or architectural changes and struggle with rare, user-defined elements. Alternatively, composing independently trained modules [3, 25, 22, 40] requires extensive fine-tuning and scales poorly. In contrast, our train-free, instance-level framework leverages LoRA adapters for fine-grained, spatially-aware concept integration.

4.2 LoRA-Based Skill Composition

Low-Rank Adaptation (LoRA) enables efficient personalization in diffusion models [35, 41]. Strategies for composing multiple LoRAs broadly fall into two categories:

Trainable Approaches: LoRAHub [12] and ZipLoRA [37] use few-shot demonstrations to learn a linear combination of LoRA weights. Others propose mixture-of-experts (MoE) [47] or hypernetworks [38, 36] to generate target weights. These methods demand additional training data and struggle in zero-shot or open-vocabulary settings.

Train-free Approaches: These methods manipulate weights or inference directly. LoRA Merge [15] uses weight-level arithmetic, while CLoRA [29] targets attention maps. LoRA Switch and Composite [46] alternate or aggregate outputs during denoising, and [48] adds frequency-based scheduling and caching. MultLFG [34] employs frequency-domain guidance but necessitates costly per-step RGB decoding, bypassing the more intuitive latent noise space. As the number of LoRAs increases, these methods often suffer from semantic conflicts and instability. Crucially, they ignore interactions with the base model, overlook domain shifts caused by fine-tuning, and are limited to text conditions.

Inspired by spatial composition techniques like CutMix [45] and token-level fusion [43], LoRAtorio applies these principles in the latent space. While resembling an MoE [16], our method is entirely train-free: it uses intrinsic cosine similarity between LoRA and base model latents for zero-shot gating, operating directly within the semantic latent space rather than relying on learned routing or image-space features.

5 Conclusion

In this paper, we present a novel, train-free approach to multi-LoRA composition through the introduction of LoRAtorio, a method grounded in intrinsic model behaviour. Motivated

by empirical observations of domain drift and latent-space divergence, our method leverages spatially-aware similarity to dynamically weight LoRA contributions at the patch level. We further propose a modification to classifier-free guidance that incorporates the base model's unconditional signal, improving robustness in out-of-domain scenarios. Extending beyond static composition, we formulate the task as one of dynamic module selection, enabling inference-time adaptability in settings where irrelevant skills are loaded to the base model. Our approach achieves state-of-the-art performance on *ComposLoRA* with CLIPScore that does not deteriorate as the number of composed adapters grows, and we further demonstrate the portability of the intrinsic-similarity principle to a rectified-flow backbone.

References

[1] Dan Biderman, Jacob Portes, Jose Javier Gonzalez Ortiz, Mansheej Paul, Philip Greengard, Connor Jennings, Daniel King, Sam Havens, Vitaliy Chiley, Jonathan Frankle, et al. Lora learns less and forgets less. *arXiv preprint arXiv:2405.09673*, 2024.

[2] Rujikorn Charakorn, Edoardo Cetin, Yujin Tang, and Robert Tjarko Lange. Text-to-LoRA: Instant Transformer Adaption, June 2025. URL `http://arxiv.org/abs/2506.06105`. arXiv:2506.06105 [cs].

[3] Yilun Du, Shuang Li, and Igor Mordatch. Compositional visual generation with energy based models. *Advances in Neural Information Processing Systems*, 33:6637–6647, 2020.

[4] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024.

[5] Weixi Feng, Xuehai He, Tsu-Jui Fu, Varun Jampani, Arjun Reddy Akula, Pradyumna Narayana, Sugato Basu, Xin Eric Wang, and William Yang Wang. Training-free structured diffusion guidance for compositional text-to-image synthesis. In *The Eleventh International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=PUIqjT4rzq7`.

[6] Zihao Fu, Haoran Yang, Anthony Man-Cho So, Wai Lam, Lidong Bing, and Nigel Collier. On the Effectiveness of Parameter-Efficient Fine-Tuning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 12799–12807, 2023.

[7] Oran Gafni, Adam Polyak, Oron Ashual, Shelly Sheynin, Devi Parikh, and Yaniv Taigman. Make-a-scene: Scene-based text-to-image generation with human priors. In *European Conference on Computer Vision*, pages 89–106. Springer, 2022.

[8] Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. CLIPScore: A reference-free evaluation metric for image captioning. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7514–7528, November 2021.

[9] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021.

[10] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[11] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1 (2):3, 2022.

[12] Chengsong Huang, Qian Liu, Bill Yuchen Lin, Tianyu Pang, Chao Du, and Min Lin. Lorahub: Efficient cross-task generalization via dynamic lora composition. In *Conference on Language Modeling*, 2024.

[13] Lianghua Huang, Di Chen, Yu Liu, Yujun Shen, Deli Zhao, and Jingren Zhou. Composer: creative and controllable image synthesis with composable conditions. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.

[14] Ziyao Huang, Fan Tang, Yong Zhang, Xiaodong Cun, Juan Cao, Jintao Li, and Tong-Yee Lee. Make-your-anchor: A diffusion-based 2d avatar generation framework. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6997–7006, June 2024.

[15] Hugging Face. Merging loras. https://huggingface.co/docs/diffusers/en/using-diffusers/merge_loras, 2024. Accessed: 2025-06-09.

[16] Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87, 1991. doi: 10.1162/neco.1991.3.1.79.

[17] 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.

[18] Tero Karras, Miika Aittala, Tuomas Kynkäänniemi, Jaakko Lehtinen, Timo Aila, and Samuli Laine. Guiding a diffusion model with a bad version of itself. In *Advances in Neural Information Processing Systems*, volume 37, pages 52996–53021, 2024.

[19] Nupur Kumari, Bingliang Zhang, Richard Zhang, Eli Shechtman, and Jun-Yan Zhu. Multi-concept customization of text-to-image diffusion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1931–1941, 2023.

[20] Black Forest Labs. Flux. https://github.com/black-forest-labs/flux, 2024.

[21] Jialu Li, Jaemin Cho, Yi-Lin Sung, Jaehong Yoon, and Mohit Bansal. SELMA: Learning and merging skill-specific text-to-image experts with auto-generated data. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=t9gNEhreht.

[22] Shuang Li, Yilun Du, Joshua B. Tenenbaum, Antonio Torralba, and Igor Mordatch. Composing ensembles of pre-trained models via iterative consensus. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=gmwDKo-4cY.

[23] Siwei Li, Yifan Yang, Yifei Shen, Fangyun Wei, Zongqing Lu, Lili Qiu, and Yuqing Yang. Lorasc: Expressive and generalizable low-rank adaptation for large models via slow cascaded learning. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 12806–12816, 2024.

[24] Kevin Lin, Zhengyuan Yang, Linjie Li, Jianfeng Wang, and Lijuan Wang. Designbench: Exploring and benchmarking dall-e 3 for imagining visual design. *arXiv preprint arXiv:2310.15144*, 2023.

[25] Nan Liu, Shuang Li, Yilun Du, Josh Tenenbaum, and Antonio Torralba. Learning to compose visual relations. *Advances in Neural Information Processing Systems*, 34:23166–23178, 2021.

[26] Xingchao Liu, Chengyue Gong, and qiang liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *The Eleventh International Conference on Learning Representations*, 2023.

[27] Davide Lobba, Fulvio Sanguigni, Bin Ren, Marcella Cornia, Rita Cucchiara, and Nicu Sebe. Inverse virtual try-on: Generating multi-category product-style images from clothed individuals. *arXiv preprint arXiv:2505.21062*, 2025.

[28] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *arXiv preprint arXiv:2211.01095*, 2022.

[29] Tuna Han Salih Meral, Enis Simsar, Federico Tombari, and Pinar Yanardag. Clora: A contrastive approach to compose multiple lora models, 2024.

[30] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pages 8162–8171. PMLR, 2021.

[31] Ziheng Ouyang, Zhen Li, and Qibin Hou. K-lora: Unlocking training-free fusion of any subject and style loras. *arXiv preprint arXiv:2502.18461*, 2025.

[32] Akshara Prabhakar, Yuanzhi Li, Karthik Narasimhan, Sham Kakade, Eran Malach, and Samy Jelassi. Lora soups: Merging loras for practical skill composition tasks. In *Proceedings of the 31st International Conference on Computational Linguistics: Industry Track*, pages 644–655, 2025.

[33] 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.

[34] Aniket Roy, Maitreya Suin, Ketul Shah, and Rama Chellappa. Multlfg: Training-free multi-lora composition using frequency-domain guidance. *arXiv preprint arXiv:2505.20525*, 2025.

[35] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22500–22510, 2023.

[36] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Wei Wei, Tingbo Hou, Yael Pritch, Neal Wadhwa, Michael Rubinstein, and Kfir Aberman. Hyperdreambooth: Hypernetworks for fast personalization of text-to-image models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6527–6536, 2024.

[37] Viraj Shah, Nataniel Ruiz, Forrester Cole, Erika Lu, Svetlana Lazebnik, Yuanzhen Li, and Varun Jampani. Ziplora: Any subject in any style by effectively merging loras. In *European Conference on Computer Vision*, pages 422–438. Springer, 2024.

[38] Donald Shenaj, Ondrej Bohdal, Mete Ozay, Pietro Zanuttigh, and Umberto Michieli. LoRA.rar: Learning to Merge LoRAs via Hypernetworks for Subject-Style Conditioned Image Generation, December 2024. URL `http://arxiv.org/abs/2412.05148`. arXiv:2412.05148 [cs].

[39] Chenyang Si, Ziqi Huang, Yuming Jiang, and Ziwei Liu. Freeu: Free lunch in diffusion u-net. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4733–4743, 2024.

[40] Enis Simsar, Thomas Hofmann, Federico Tombari, and Pinar Yanardag. Loraclr: Contrastive adaptation for customization of diffusion models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 13189–13198, 2025.

[41] Kihyuk Sohn, Nataniel Ruiz, Kimin Lee, Daniel Castro Chin, Irina Blok, Huiwen Chang, Jarred Barber, Lu Jiang, Glenn Entis, Yuanzhen Li, et al. Styledrop: text-to-image generation in any style. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, pages 66860–66889, 2023.

[42] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021.

[43] Zirui Wang, Zhizhou Sha, Zheng Ding, Yilin Wang, and Zhuowen Tu. Tokencompose: Text-to-image diffusion with token-level supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8553–8564, 2024.

[44] Yang Yang, Wen Wang, Liang Peng, Chaotian Song, Yao Chen, Hengjia Li, Xiaolong Yang, Qinglin Lu, Deng Cai, Boxi Wu, et al. Lora-composer: Leveraging low-rank adaptation for multi-concept customization in training-free diffusion models. *arXiv preprint arXiv:2403.11627*, 2024.

[45] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6023–6032, 2019.

[46] Ming Zhong, Shuohang Wang, Yadong Lu, Yizhu Jiao, Siru Ouyang, Donghan Yu, Jiawei Han, Weizhu Chen, et al. Multi-lora composition for image generation. In *Transactions on Machine Learning Research*, 2024.

[47] Jie Zhu, Yixiong Chen, Mingyu Ding, Ping Luo, Leye Wang, and Jingdong Wang. MoLE: Enhancing Human-centric Text-to-image Diffusion via Mixture of Low-rank Experts. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 29354–29386. Curran Associates, Inc., 2024. URL `https://proceedings.neurips.cc/paper_files/paper/2024/file/3415a8f8127d5b0ceb7fd321180b1954-Paper-Conference.pdf`.

[48] Xiandong Zou, Mingzhu Shen, Christos-Savvas Bouganis, and Yiren Zhao. Cached multi-lora composition for multi-concept image generation. In *13th International Conference on Learning Representations*, 2025.

A Ethics Statement

This work examines the capabilities of generative AI models, including those enhanced with community-provided LoRAs. While generative tools offer valuable opportunities for creative and technical innovation, they also carry significant risks, including misuse for deceptive content, reinforcement of harmful biases, and uncertainty around authorship and licensing.

We do not condone the misuse of generative models, including for misinformation, harassment, or any activity that infringes on the rights of others. This work is not licensed or intended for commercial or for-profit use. We encourage future users and researchers to carefully consider the ethical and legal implications of models or data.

B Theoretical Motivation for Similarity-Based Weighting.

LoRA introduces a low-rank update to a weight matrix $W \in \mathbb{R}^{d\times k}$ in the form $\Delta W = AB$, where $A \in \mathbb{R}^{d\times r}$, $B \in \mathbb{R}^{r\times k}$, and $r \ll \min(d, k)$ [11]. This constrains the update to lie in a low-dimensional subspace of the weight space, limiting the directions in which the model can adapt. Such low-rank adaptation has been shown to improve parameter efficiency and mitigate catastrophic forgetting [1].

More precisely, the LoRA update acts on inputs $x \in \mathbb{R}^k$ by first projecting via A, $Ax \in \mathbb{R}^r$, then mapping back to output space via B. The effective input subspace to which the adapter responds is the row space of A, i.e., inputs x for which $Ax \neq 0$. For inputs x' approximately orthogonal to this subspace, $Ax' \approx 0$ and thus

$$\Delta W x' = (Ax')B \approx 0, \tag{9}$$

implying

$$Wx' + \Delta W x' \approx Wx'. \tag{10}$$

Hence, the LoRA adapter has a negligible effect on inputs lying outside its learned subspace, which often correspond to out-of-distribution (OOD) inputs, and subsequent non-linearities in deep learning models further mitigate the effect of LoRAs. Consequently, the latent outputs of the LoRA-augmented model and the base model are similar for OOD inputs. This motivates using the cosine similarity between their latent outputs as a proxy for the adapter's confidence or relevance: high similarity indicates that the adapter is inactive or uncertain (OOD), whereas lower similarity suggests in-distribution behaviour where the adapter actively modifies the model output. This observation underpins our use of cosine similarity in LoRAtorio.

B.1 Intuition Behind Similarity-Based Weighting

To supplement the theoretical motivation above, we provide an intuitive explanation for our patch-level cosine similarity mechanism. Prior works have established that diffusion features correlate strongly with the spatial layout of the generated image. For a fixed initial noise seed, different LoRA adapters maintain a consistent "structural blueprint" that closely mirrors the base model, particularly when the text condition is out-of-distribution for the adapter.

LoRAtorio leverages this spatial consistency. When a specific LoRA is highly relevant to a given spatial patch (e.g., a character's face), it actively modifies the local latent representation, causing a significant, meaningful divergence from the base model's prediction. The SoftMin function subsequently assigns a higher weight to these divergent (low-similarity) patches. This localized gating mechanism prevents the "blurred average" phenomenon observed in simple weight-merging techniques by ensuring that only the most spatially relevant experts contribute to specific regions of the final image.

C Preliminary Motivational Experiments

To further substantiate the visual intuition presented in Figure 1a, and to quantitatively support Observation 1 and Observation 2, we perform an empirical analysis of the latent

space distributions. While visualisations such as t-SNE provide an intuitive understanding of the domain shift caused by Low-Rank Adaptation (LoRA), we supplement this evaluation with quantitative metrics.

We isolate the unconditional noise predictions from the base model and the LoRA-adapted models to measure the extent to which specialised LoRAs diverge from the base distribution. For this experiment, we extract the unconditional latent noise predictions $e_\theta(z_0, 0)$ and $e_{\theta_i}(z_0, 0)$ across 100 random seeds. To make the comparison computationally tractable while preserving the core structural variances, we apply Principal Component Analysis (PCA) to project the high-dimensional latents into a 2D space that retains 95% of the variance.

Over these estimated 2D density distributions, we calculate the Kullback-Leibler (KL) divergence from the base model distribution P to the ith LoRA distribution Q_i:

$$D_{KL}(P \parallel Q_i) = \int P(x) \log \left(\frac{P(x)}{Q_i(x)} \right) dx \tag{11}$$

As shown in Table 5, highly specialised domains—particularly characters and specific artistic styles—exhibit a significant KL divergence from the base model. Backgrounds and generic objects, which are inherently closer to the diverse pre-training data of the base model, show comparatively lower divergence.

Table 5: KL Divergence of single LoRA adapters against the base model.

LoRA Model	**KL Divergence**
Character 1	34.81
Character 2	35.09
Character 3	36.15
Style 1	36.44
Style 2	36.31
Object 1	35.52
Object 2	21.49
Background 1	21.34
Background 2	9.43

D Experimental Results

Table 6: ClipScores for *ComposLoRA* on anime and reality subsets.

(a) Anime – Static Modules

Model	$N=2$	$N=3$	$N=4$	$N=5$	Avg.
Merge	35.136	35.421	34.164	32.636	34.339
Switch	35.285	35.482	34.532	34.148	34.861
Composite	34.343	34.378	34.161	32.936	33.955
LoraHub	35.316	35.525	34.476	33.885	34.801
Switch-A	35.705	35.912	35.661	34.479	35.439
CMLoRA	35.556	35.555	35.791	35.691	35.648
MultLFG	**36.720**	36.130	36.450	36.220	36.380
LoRAtorio	36.156	**36.930**	**36.864**	**36.162**	**36.528**

(b) Reality – Static Modules

Model	$N=2$	$N=3$	$N=4$	$N=5$	Avg.
Merge	32.316	32.857	32.633	32.091	32.474
Switch	35.502	34.731	34.424	32.801	34.365
Composite	35.804	33.786	35.443	32.228	34.315
LoraHub	36.045	34.729	35.463	33.084	35.412
Switch-A	35.196	34.854	34.694	32.252	34.249
CMLoRA	35.559	35.842	34.501	33.588	34.873
MultLFG	**36.420**	36.120	35.910	35.620	36.018
LoRAtorio	34.316	**35.922**	**37.408**	**37.090**	**36.184**

(c) Anime – Dynamic Modules

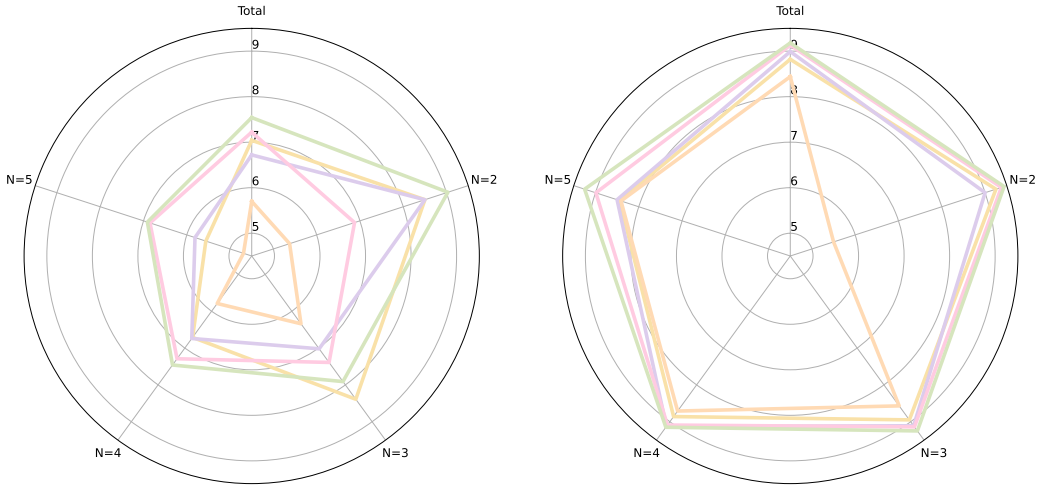
	$N=2$	$N=3$	$N=4$	$N=5$	Avg.
LoRAtorio	**35.328**	**35.931**	**36.332**	**36.184**	**35.944**
Naïve	35.014	34.927	34.384	33.809	34.534
Merge	30.953	30.767	30.352	30.488	30.640

(d) Reality – Dynamic Modules

	$N=2$	$N=3$	$N=4$	$N=5$	Avg.
LoRAtorio	33.858	**35.194**	**36.627**	**37.871**	**35.888**
Naïve	**35.014**	34.927	34.384	33.809	34.534
Merge	23.381	23.534	23.693	24.055	23.666

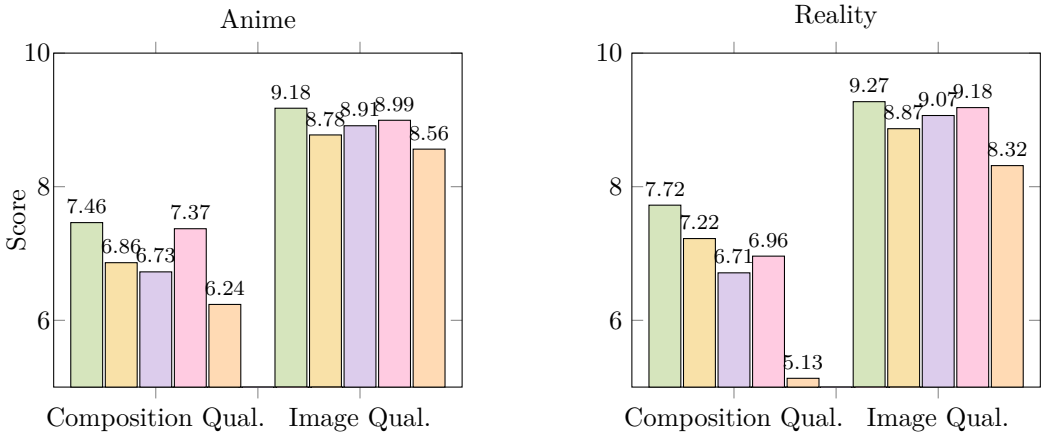
Further to the main experimental results in Section 3, we show CLIPScore for the subsets of *ComposLoRA* in Table 6 for anime and reality subsets in both static and dynamic module

settings. LoRAtorio maintains strong performance in both subsets, having the highest weighted average score compared to previous SoTA. This is consistent for both static and dynamic module selection. We observe similar trends in performance within each subset for the number of LoRAs included, with LoRAtorio clearly outperforming other works on average. As expected, we also see weight merge collapsing in the dynamic module setting.



(a) Composition Quality by N

(b) Image Quality by N



(c) Breakdown by Image Style

Figure 5: Composition and Image Quality scores of LoRAtorio against previous SoTA on *ComposeLoRA*, by number of LoRA's included (a-b) and image style (c).

Further to the ClipScores, we present the GPT-4V evaluation results by number of LoRAs included and by sub-set in 5. LoRAtorio maintains robust performance in all scenarios, showing strong composition and image quality. Finally, we include the win rate of our method by number of LoRAs included, showing SoTA performance, particularly as N increases in Figure 6.

[2]CLIPScore for the images in each row is identical, including autoguidance samples highlighting the necessity of visual inspection and qualitative evaluation.
Row 1:32.552, Row 2: 31.810, Row 3: 31.763, Row 4: 32.653

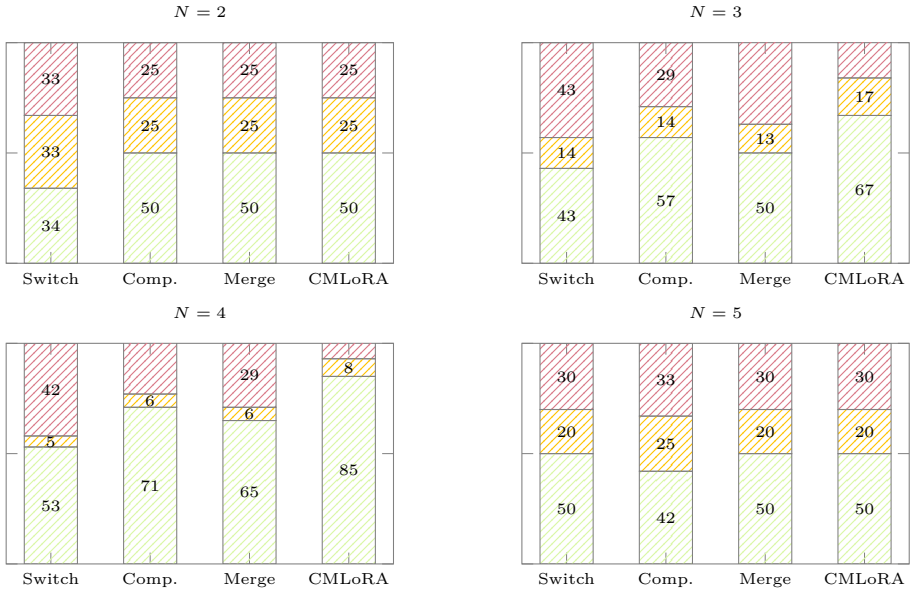


Figure 6: Win/Tie/Loss for LoRAtorio compared to previous SoTA across different number of LoRAs (N) .

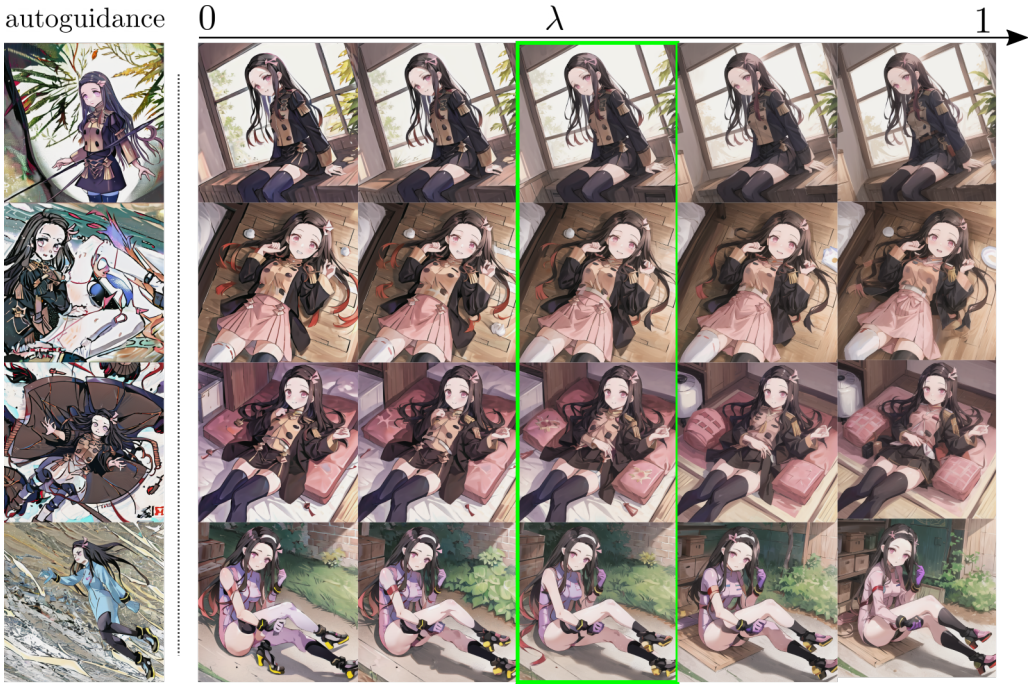


Figure 7: Qualitative comparison of LoRAtorio's re-centering guidance across different values of λ, evaluated against auto-guidance [18]. The figure illustrates the impact of varying λ on image coherence, identity preservation, and visual quality.[2]

D.1 Ablation Study

To show the effect of LoRAtorio's components, we perform an ablation study as shown in Table 7, using CLIPScore as an evaluation metric. Note that CLIPScore is unable to capture the composition of aesthetic quality, so the final set of hyperparameters is selected as a combination of CLIPScore and empirically through visual inspection of output. Specifically, we compute the CLIPScore of generated images using the distance of the entire image instead of individual patches, with a constant $\tau = 1$ and without our re-centering method. The localised activation of LoRAs through the tokenisation of the latent space has the greatest impact in terms of CLIPScore, which is somewhat expected as more elements can be integrated and thus aligned in clip space. We also compare the effect of patch size on the performance of LoRAtorio and see that a more fine-grained composition results in higher CLIPScore, although the performance is relatively robust. As such, we select the patch size according to the finest-grained concept in the prompt, since localised concepts (e.g., characters) benefit from finer patches, whereas global LoRAs (e.g., style) are largely insensitive to patch size (Figure 9).

Table 7: Ablation study of our method on *ComposLoRA* Anime subset, using CLIPScore.

(a) LoRAtorio Components

	$N = 2$	$N = 3$	Avg.
LoRAtorio	36.156	**36.930**	**36.543**
w/o $\phi(\cdot)$	34.306	33.967	34.137
w/o τ	35.948	36.690	36.319
w/o Re-centering	**36.477**	36.586	36.532

(b) Patch Size

	$N = 2$	$N = 3$	Avg.
2×2	**36.156**	**36.930**	**36.543**
4×4	36.025	36.475	36.250
8×8	35.852	36.423	36.138
16×16	35.744	36.003	35.874

In addition, we conduct a qualitative comparison of LoRAtorio's re-centering guidance with auto-guidance [18], and evaluate performance across different values of the weighting parameter λ, shown on Figure 7, as CLIPScore alone does not capture identity preservation, compositionally or other qualitative elements. More specifically, we show in Figure 7 that the CLIPScore is consistent for all selected images, thus reiterating that it should be used as a metric of generic object inclusion, not instance fidelity or qualitative score. Since the base model is essentially "a bad version" of the LoRA-augmented model, auto-guidance serves as a natural baseline for assessing our re-centering approach. Notably, we find that combining LoRAtorio with auto-guidance fails to produce coherent images. We hypothesise this is due to a difference in data distribution, a prerequisite for auto-guidance. Similarly, when $\lambda = 0$ – where the unconditioned score corresponds to that of the base model – we observe strong identity preservation, but the resulting images exhibit excessive saturation and appear unnatural. Conversely, setting $\lambda = 1$ results in some loss of identity and a blending of concepts. To balance these effects, we empirically select $\lambda = 0.5$ for all experiments, for simplicity. We further corroborate this choice with a pairwise LLM-as-a-judge evaluation over $\lambda \in [0, 1]$ on character, object and clothing concepts, where $\lambda = 0.5$ is preferred in 88% of comparisons and the adjacent $\lambda = 0.25$ in the remainder.

E Temporal Analysis of Similarity-based Weighting

Our similarity-based weighting mechanism is designed as a proxy of the relative confidence of each LoRA adapter with respect to the base model. Empirically, we observe that the cosine similarity between LoRA-augmented outputs and the base model varies non-uniformly across denoising steps, depending on the LoRA employed.

This temporal asymmetry aligns with prior findings in diffusion literature [39, 46, 48], which show that different semantic attributes emerge at different stages of the denoising process. However, we observe that the variation within LoRAs of the same type (e.g. clothing or style) is too vast for universal and concrete conclusions on the order of activations. We believe this to be due to different LoRA training and configuration, further explored in Appendix F

To validate this behaviour, we analyse the evolution of the similarity matrix Ω^t over time, aggregated across the latent representation. As shown in Figure 8, the pattern is not very

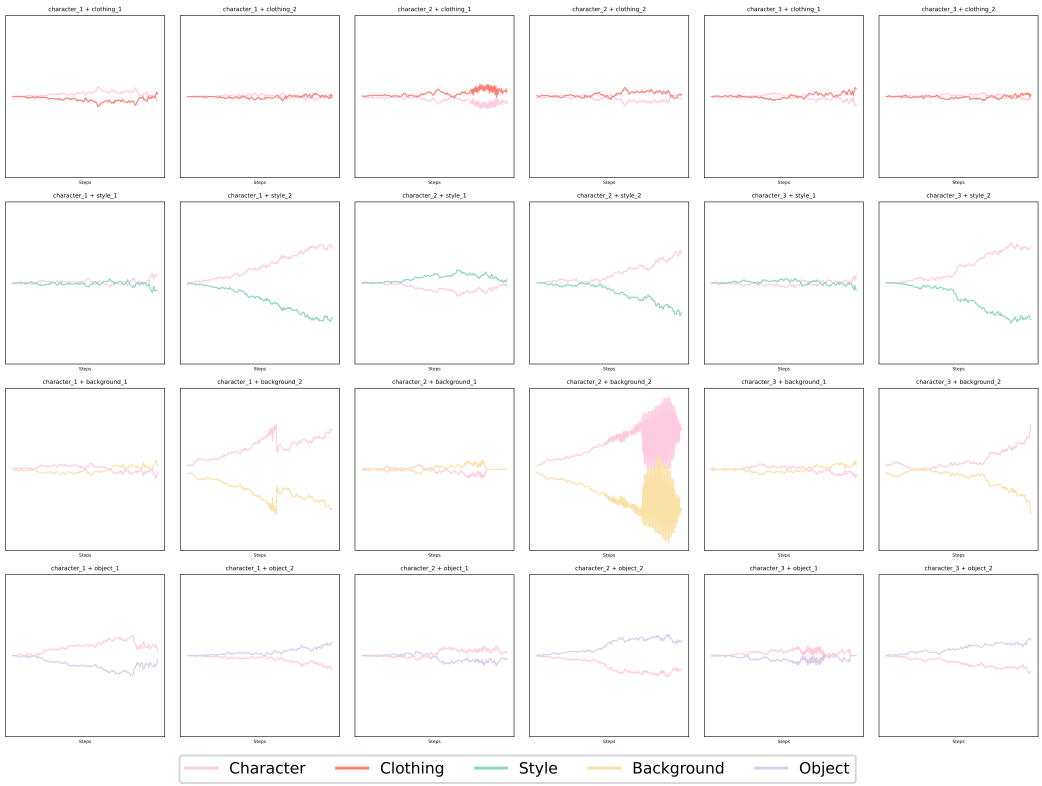


Figure 8: Average values of Ω^t over denoising process, for the entire image in the *ComposLoRA* testbed for $N = 2$.

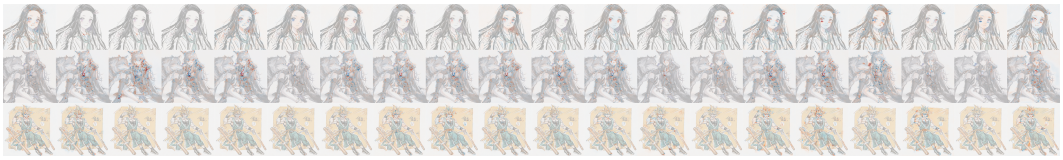


Figure 9: Ω^t map for character overlayed over the final generated images across timesteps, in Character + Style generation. As style has more global effect, we see clearly the effect of the character LoRA on the predicted noise through the Ω^t heatmap (high effect is red, no effect is transparent).

consistent for any of the element groups. As such, methods relying on guidance based on the type of LoRA used ignore this intrinsic proxy for confidence completely. However, because different elements vary in spatial extent, a naïve global aggregation would disproportionately favour larger elements – particularly in early steps that affect the trajectory of the denoising process [46]. This motivates our use of the spatial tokenisation function ϕ, which enables fine-grained, patch-level weighting and ensures that larger background or clothing regions do not overshadow smaller but semantically important regions (e.g., smaller objects). A visualisation of the patchwise similarity over the diffusion process can be seen in Figure 9.

Spatial Coherence: To quantify whether independent per-patch weighting introduces spatial fragmentation, we compute the normalised total variation of the Ω^t maps. We observe that it remains approximately 0.4 throughout the structure-forming window, compared to approximately 1.5 for a fragmented reference, obtained by randomly permuting the same weight values across spatial locations. The weight of each adapter is concentrated in a single coherent region, and the total variation only rises in the late steps as fine texture resolves. Furthermore, as the weighting is applied to the predicted noise in the low-resolution latent

space rather than in pixel space, any patch boundaries are smoothed by the remaining denoising steps and the VAE decoder.

Region-level Alignment: We further evaluate whether concepts are routed to their own regions using masked image-text alignment. Specifically, we segment the region of each concept with a grounding model, crop it, and compute its CLIPScore against both its own concept phrase and the phrase of a competing concept. We observe that a spatially distinct object concept aligns with its own phrase far more strongly than with the character phrase, with a within-region minus cross-region CLIPScore gap of +9.6. Note that we report this gap rather than a hard mIoU, as *ComposLoRA* compositions are entangled by construction (e.g., a character wears the clothing and holds the object). As such, a hard mIoU would measure the unavoidable overlap of the benchmark rather than the quality of the gating.

F Extended Limitations and Failure Mode Analysis

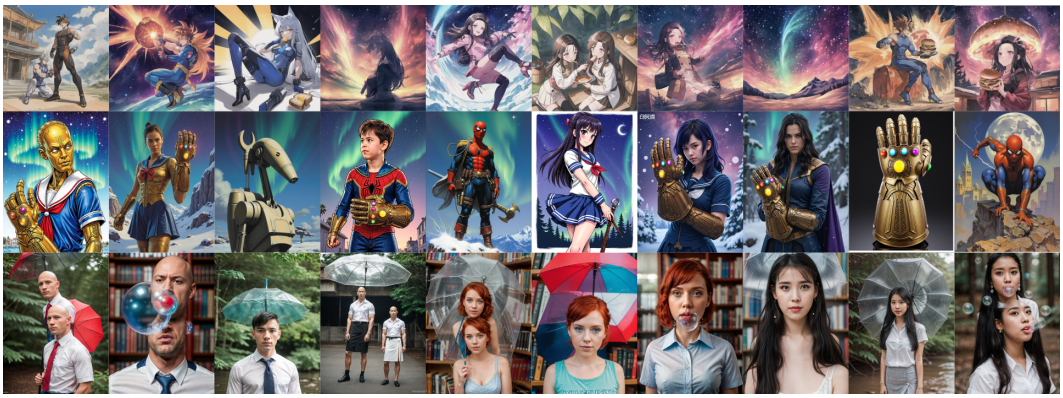


Figure 10: Examples of failure cases of LoRAtorio on *ComposLoRA* anime (top), Flux (mid) and *ComposLoRA* reality (bottom) test beds.

One key limitation of our method is the computational cost. While the intrinsic nature of LoRAtorio allows for a better understanding of the generation process and shows competitive results, the computational cost increases linearly with every additional LoRA. This limitation is identified by previous works manipulating the latent space instead of the weights [46]. This is especially true in the open-vocabulary setting where all available LoRA adapters are loaded. Potential future directions to address these limitations include exploring the subspace at an earlier stage (i.e. based on early-layer similarity, which we have not explored in the scope of this work), so that pruning or TopK can be implemented before obtaining the latent denoised output. Furthermore, model parallelisation may increase the speed of inference by estimating denoised outputs on different GPUs.

A primary challenge for any training-free composition method is managing heterogeneous adapter quality. Because LoRAs are trained independently by the community on varying datasets with undocumented hyperparameter configurations, their output magnitudes and latent topologies can differ drastically.

Spurious Divergence from Low-Quality LoRAs: Our core assumption is that latent divergence from the base model acts as a proxy for LoRA relevance and confidence. However, an overfitted or poorly trained LoRA can produce inherently "noisy" latent predictions. The SoftMin gating mechanism may misinterpret this chaotic deviation as high-confidence activation, assigning a large weight to a misaligned adapter. This failure mode manifests visually as severe concept confusion or structural collapse.

Quantitative Indicators of Failure: As demonstrated in the quantitative analysis in Section C, these failure modes can be anticipated by mapping KL-Divergence against CLIP-fidelity. Failures typically occur when a single adapter exhibits an extremely high KL-divergence from the base model but yields a simultaneously low CLIPScore. This combination indicates that the LoRA has significantly warped the latent space away from

the base distribution without successfully capturing the target semantic concept, leading to inevitable clashes when composed with other adapters.

Empirical Validation with Corrupted Adapters: To validate this indicator, we construct two deliberately corrupted adapters and compare them to five clean *ComposLoRA* adapters. Specifically, we use one over-fitted adapter, trained to memorise a single image, and one domain-mismatched adapter, trained on one concept but registered under the trigger of a different concept. For each adapter, we measure its divergence from the base model and its CLIP-fidelity to the concept it claims. We compute the divergence as the mean relative difference between the noise predictions of the adapter and the base model under the trigger of the adapter, which is an empirical proxy for the KL-divergence above. As shown in Table 8, the over-fitted adapter is identified by its divergence, which is roughly two to three times that of any clean adapter. In contrast, the domain-mismatched adapter falls in the high-divergence, low-fidelity region predicted above. Left unmitigated, SoftMin gating assigns the corrupted adapter the top patch-weight share in 17% of the probed prompts, and an above-uniform share in 83%. However, as both corrupted adapters are separable in a single cached forward pass, they can be screened from the candidate pool before composition.

Table 8: Divergence from the base model and CLIP-fidelity for clean and deliberately corrupted adapters.

Adapter group	Divergence from base	CLIP-fidelity
Clean (5 adapters)	0.008–0.022	25–36
Over-fitted	0.050	35
Domain-mismatched	0.082	16.6

Our method assumes that LoRA adapters are trained on reasonably well-aligned and semantically coherent datasets. In practice, however, LoRA quality can vary significantly, particularly when sourced from community repositories such as CivitAI, where training data, objectives, and preprocessing pipelines are often undocumented or inconsistent. This variability can undermine the reliability of any train-free approach. Moreover, the LoRAs used during inference are heterogeneous in terms of optimal hyperparameters (e.g., guidance scale, LoRA scale), and treating them uniformly may inadvertently bias the composition toward certain adapters, especially those with more aggressive or dominant activations. While we expect better performance when LoRAs are trained under similar conditions, such alignment is rarely guaranteed in user-driven settings. One potential mitigation strategy in real-life applications is to incorporate a lightweight pre-filtering step to assess LoRA quality before inclusion. Alternatively, metadata-based heuristics (e.g., dataset size, training steps, or CFG guidance scale) could be used to cluster or filter LoRAs. Although these approaches are not explored in this work beyond the divergence–fidelity screening above, they represent promising directions for improving robustness in real-world deployments. Finally, as all LoRAs in the *ComposLoRA* and Flux testbeds are sourced from CivitAI without access to training details, we emphasise that all results should be interpreted in light of this uncertainty.

Finally, we note that the quality of images is affected by the base model. Figure 10 shows examples of fail cases of LoRAtorio for all three base models. We observe that the Stable Diffusion backbone (top and bottom rows) exhibits more instances of additional limbs or duplicate characters compared to the Flux backbone (middle row), where most failure cases are attributed to concept confusion. As such, expanding the test bed to more backbones is essential in dissecting base model vs method limitations.

G Flux Testbed and Implementation Details

For experiments on Flux, we select the LoRAs described in Table 9, following a selection process similar to *ComposLoRA*. All LoRAs used in the Flux experiments are publicly available through CivitAI. We select LoRAs for three characters, two clothing items, two styles, two objects and one background.

Table 9: Details of LoRA adapters used in Flux experiments.

LoRA	Category	Trigger	Source
Yennefer of Vengerberg	Character	Yennefer	Link
The amazing Spiderman	Character	Spider-Man, Peter Parker	Link
B1 Battle Droid	Character	7-B1 droid	Link
Star Wars imperial officer uniform	Clothing	Wearing an imperial officer IMPOFF uniform	Link
Japanese school uniform - sailorfuku	Clothing	wearing a japanese school uniform sailorfuku serafuku sailor suit	Link
Frank Frazetta Style Oil Painting	Style	in the style of Frank Frazetta fantasy oil painting	Link
Engraving Style	Style	in engraving style	Link
Infinity Goblet	Object	with a glove like Infinity Gauntlet	Link
Crescent Wrench	Object	with a Crescent Wrench	Link
Northern Lights	Background	with Northern lights style background	Link

G.1 Architecture Adaptations and Baseline Selection

To our knowledge, we are the first to extend multi-LoRA inference-time composition to rectified-flow architectures, specifically targeting the transformer-based Flux.1-dev model. This required specific adaptations to the LoRAtorio framework:

Omission of Tokenization and CFG Re-centering: Unlike Stable Diffusion v1.5, Flux relies on a transformer-based architecture where the latent space is natively tokenised; therefore, we apply our weighting directly to the denoised network outputs without requiring an explicit patch-tokenization step. Furthermore, Flux does not employ traditional Classifier-Free Guidance (CFG)—where inputs are passed twice through the network—but instead embeds control signal parameters directly within the cross-attentions. Consequently, the CFG adjustment described in Equations 1 and 2 is structurally incompatible with Flux, and the re-centering step is naturally omitted. Equations 3 through 6 remain fully applicable.

Baseline Selection: Adapting existing multi-LoRA methods to rectified-flow architectures requires extensive re-engineering. We selected Naïve composition and LoRA Merge as our primary baselines for the Flux testbed because they are the only training-free, multi-LoRA methods natively supported out-of-the-box for this architecture.

H GPT-4V Evaluation Interface

For the GPT-4V evaluation, we use the prompt and method of the original *ComposLoRA* [46]. Specifically, we do pairwise comparisons of our method against previous works. The comparison is run twice for each pair, switching the order of images to account for any bias induced from ordering. The scores are then averaged. Pseudo-code of the evaluation can be seen in Figure 11. The prompt used in the evaluation can be seen in Figure 12.

Table 10: Comparison of Multiply-Accumulate Operations (MACs) and qualitative latency estimates under different N values.

(a) MACs (in GigaOps) for different methods.

	$N=2$	$N=3$	$N=4$	$N=5$
LoRAtorio	1090.863	1102.721	1125.570	1132.123
CMLoRA[a]	912.350	1223.486	1358.518	1570.335
Switch-A[a]	734.053	730.914	739.322	731.811
LoraHub[a]	789.770	834.613	924.299	946.721
Composite[a]	1401.066	2169.199	2892.266	3615.333
Switch[a]	734.053	730.914	739.322	731.811
Merge[a]	789.770	834.613	924.299	946.721

(b) Qualitative latency estimates (seconds).

	$N=2$	$N=3$	$N=4$	$N=5$
LoRAtorio	61	85	91	122
Merge[b]	20	21	22	24
Switch[b]	16	18	19	20
Composite[b]	60	70	76	90
MultLFG[b]	90	140	180	230

[a] As reported by [48]
[b] As reported by [34]

```python
def evaluate():
    image_n = 196 # number of images to evaluate
    gpt4v = GPT4V()
    # Load images
    image_path = "images"
    images = []
    for i in range(0, image_n + 0):
        cur_image = encode_image(join(image_path, f"{i}.png"))
        images.append(cur_image)

    # Load prompts usd to generate the images
    prompts = []
    with open("image_info.json") as f:
        image_info = json.loads(f.read())
    for i in range(len(image_info)):
        cur_prompt = "\n".join(image_info[i]["prompt"])
        prompts.append(cur_prompt)

    # Comparative evaluation
    gpt4v = GPT4V()
    gpt4v_scores = [{} for _ in range(image_n)]
    # i:     method 1
    # i + 1: method 2
    for i in tqdm(range(0, image_n, 2)):
        method1_image     = images[i]
        method2_image    = images[i + 1]

        cur_prompt = get_eval_prompt(prompts[i])

        compare_images(method1_image, method2_image, "method_1", "method_2", gpt4v, cur_prompt)
        compare_images(method2_image, method1_image, "method_2", "method_1", gpt4v, cur_prompt)
```

Figure 11: Pseudo-code of GPT-4V Evaluation. For each pair of images, the comparison is run twice to account for bias in presentation order.

I Computational Cost Analysis and Dynamic Top-k Selection

In the dynamic module selection setting, LoRAtorio is tasked with selecting the most relevant adapters from a large pool without prior knowledge of the prompt's requirements.

Top-k Configuration: We clarify that the top-k selection operates locally on the Ω^t matrix, which has dimensionality $N \times P$ (where N is the number of loaded LoRAs and P is the number of spatial patches). For all dynamic experiments, we explicitly set $k = C$, where C is the number of distinct semantic concepts requested in the text prompt, allowing for a fair comparison against the static setting.

Computational Cost: Achieving true zero-shot dynamic selection requires evaluating the denoising network for all N loaded adapters at each timestep to compute the similarity matrix Ω^t and rank the outputs. While this increases the per-step memory footprint and computational overhead linearly with respect to N, Table 10 demonstrates that the total Multiply-Accumulate Operations (MACs) and inference latency remain highly competitive.

While LoRAtorio demonstrates competitive performance and interpretability, its computational cost increases linearly with the number of active LoRA modules. This is a direct consequence of its design, which composes multiple LoRAs simultaneously in the latent space. In contrast, methods like Switch or Merge maintain a relatively constant cost by activating only a subset or a merged representation of LoRAs. This limitation aligns with prior observations in latent-space manipulation approaches [46]. The cost becomes particularly significant in dynamic settings, where all available LoRA adapters may be loaded concurrently. However, we also note that the MACs of LoRAtorio do not differ by orders of magnitude compared to previous works; in fact, we see that they are comparable, thus our method does not

```
I need assistance in comparatively evaluating two text-to-image models based on their ability to compose different elements into a single image. The elements and their key features are as follows:
    <IMAGE_1> <IMAGE_2> <PROMPT>

Please help me rate both given images on the following evaluation dimensions and criteria:

Composition quality:
    - Score on a scale of 0 to 10, in 0.5 increments, where 10 is the best and 0 is the worst.
    - Deduct 3 points if any element is missing or incorrectly depicted.
    - Deduct 1 point for each missing or incorrect feature within an element.
    - Deduct 1 point for minor inconsistencies or lack of harmony between elements.
    - Additional deductions can be made for compositions that lack coherence, creativity, or realism.


Image quality:
    - Score on a scale of 0 to 10, in 0.5 increments, where 10 is the best and 0 is the worst.
    - Deduct 3 points for each deformity in the image (e.g., extra limbs or fingers, distorted face, incorrect proportions).
    - Deduct 2 points for noticeable issues with texture, lighting, or color.
    - Deduct 1 point for each minor flaw or imperfection.
    - Additional deductions can be made for any issues affecting the overall aesthetic or clarity of the image.


Please format the evaluation as follows:

For Image 1:

[Explanation of evaluation]

For Image 2:

[Explanation of evaluation]

Scores:

Image 1: Composition Quality: [score]/10, Image Quality: [score]/10

Image 2: Composition Quality: [score]/10, Image Quality: [score]/10

Based on the above guidelines, help me to conduct a step-by-step comparative evaluation of the given images. The scoring should follow two principles:
    1. Please evaluate critically.
    2. Try not to let the two models end in a tie on both dimensions.
```

Figure 12: GPT-4V prompt

introduce a significant cost-performance trade-off compared to previous works. In Table 10b, we also see a comparison of LoRAtorio against reported inference latency in seconds. Our method has comparable latency to LoRA-composite and is significantly faster than MultLFG. Furthermore, restricting the gating over all N candidates to an early window of K steps reduces the dynamic cost from $\mathcal{O}(N \cdot T)$ towards $\mathcal{O}(N \cdot K + C \cdot T)$, and the per-adapter passes can be parallelised across devices. For web-scale adapter libraries, we expect a retrieval-stage pre-selection (e.g., based on metadata) to be applied before LoRAtorio, which then composes over the loaded candidate set.

Exploratory Threshold-based Selection: We further explore a zero-shot variant that does not assume a known k. Specifically, we load all candidate adapters and accumulate the aggregate gating signal of each adapter over a short window of denoising steps, i.e., the extent to which its predicted noise diverges from the base model across patches. We compute the same signal under the empty prompt as a per-adapter baseline, with the base model acting as a null adapter. We then subtract this baseline and retain every adapter whose calibrated signal exceeds a single global margin δ. The margin is fixed once on a small calibration split and applied unchanged to all prompts, so that the number of concepts C is never provided. We evaluate selection directly as retrieval against the ground-truth concept set of each prompt, and compare against a known-k oracle that selects the top-C adapters by the same signal, as well as an always-all-N baseline that retains every loaded adapter. As shown in Table 11a, the threshold rule matches the known-k oracle on a preliminary *ComposLoRA* evaluation, without access to C. When the entire adapter pool is loaded, the threshold rule exceeds both baselines in F1 for two- and three-concept compositions (Table 11b). We note that its high recall ensures that relevant concepts are rarely dropped, while admitted irrelevant adapters are suppressed by the subsequent per-patch SoftMin gating.

Table 11: Exploratory retrieval results for threshold-based (k-free) dynamic selection.

(a) Preliminary *ComposLoRA* evaluation.

Rule	Precision	Recall	F1
Threshold (no C)	0.75	0.75	0.75
Known-k oracle (top-C)	0.75	0.75	0.75
Always-all-N	0.50	1.00	0.67

(b) Full adapter pool loaded.

Rule	Precision	Recall	F1
Threshold (no C)	0.40	0.83	**0.54**
Known-k oracle (top-C)	0.25	0.25	0.25
Always-all-N	0.23	1.00	0.37

J Qualitative Comparison

Examples comparing our method against baselines qualitatively for the SD1.5 base model can be seen in Figure 13 and Figure 14. LoRAtorio performs competitively, exhibiting fewer concept clashes and reduced vanishing of key attributes compared to previous SoTA. In the dynamic selection setting, we observe that Merge collapses, often producing non-legible or incoherent images, especially in the more complex reality subset. In contrast, LoRAtorio reliably selects the most relevant LoRA modules, resulting in sharper, more coherent generations, with clearly recognisable concepts.

Examples comparing our method against baselines for the Flux base model can be seen in Figure 15, Figure 16, and Figure 17. LoRAtorio shows lower concept confusion compared to merge. This is particularly obvious in the case of the Dynamic module selection setting, where the image quality severely deteriorates with multiple LoRAs. Even though Flux is a stronger model and thus generates more legible images compared to SD1.5 in the dynamic setting, the difference in quality compared to LoRAtorio and Naïve is substantial.

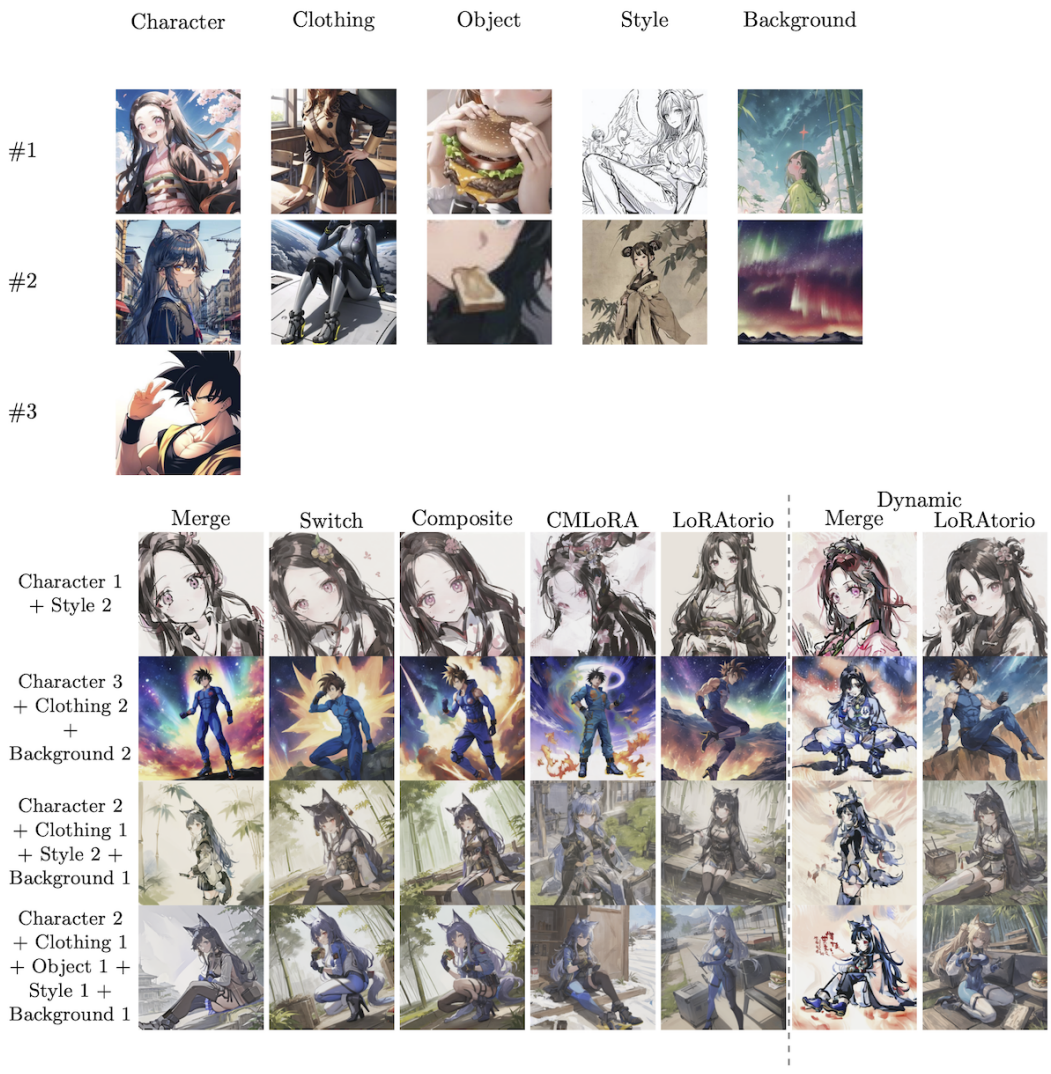


Figure 13: Images generated with N LoRA candidates (L1 Character, L2 Clothing, L3 Style, L4 Background and L5 Object) across our proposed framework and baseline methods using SD1.5 base model on the anime subset of *ComposLoRA*.



Figure 14: Images generated with N LoRA candidates (L1 Character, L2 Clothing, L3 Style, L4 Background and L5 Object) across our proposed framework and baseline methods using SD1.5 base model on the reality subset of *ComposLoRA*.



Figure 15: Images generated with N LoRA candidates (L1 Character, L2 Clothing, L3 Style, L4 Background and L5 Object) across our proposed framework and baseline methods using Flux base model.



Figure 16: Images generated with N LoRA candidates (L1 Character, L2 Clothing, L3 Style, L4 Background and L5 Object) across our proposed framework and baseline methods using Flux base model.

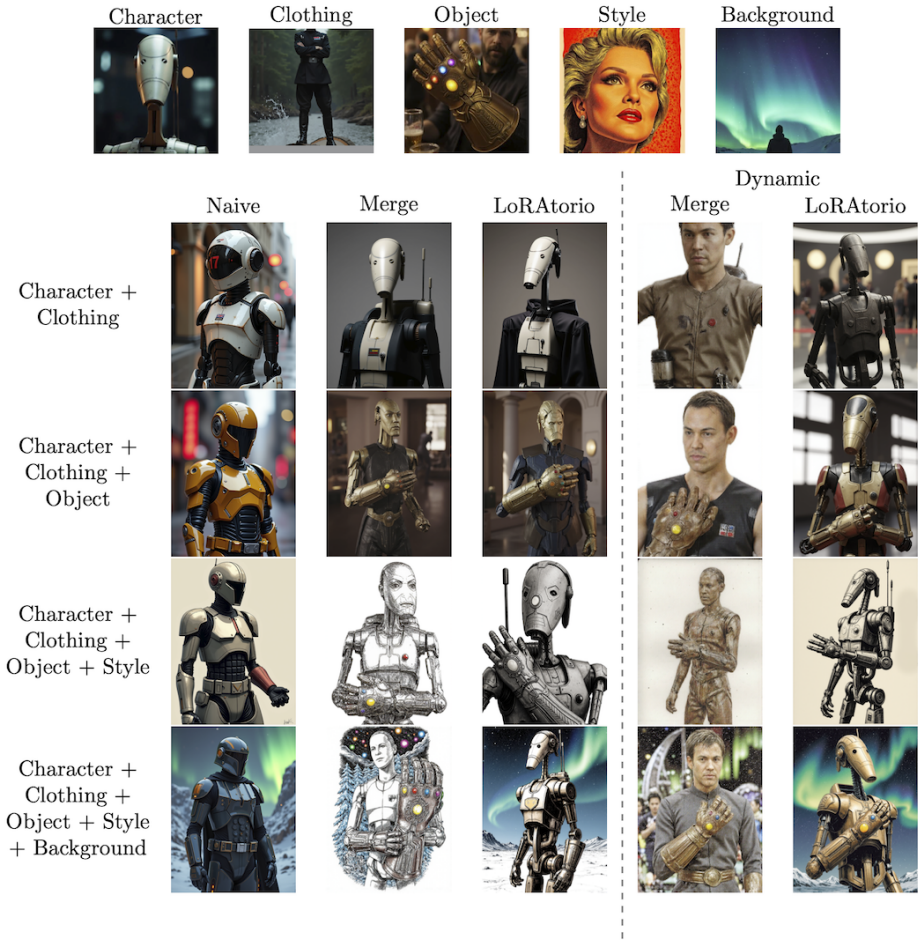


Figure 17: Images generated with N LoRA candidates (L1 Character, L2 Clothing, L3 Style, L4 Background and L5 Object) across our proposed framework and baseline methods using Flux base model.

K Human Evaluation Interface

For the human qualitative evaluation, we follow the framework of [48] along four qualitative axes. For each combination, we provide a set of reference images and the output of the anonymised methods. Samples of the instructions and survey, as shown to human experts, are presented below. We use three human experts to evaluate our method against previous SoTA. The annotators have an average Cohen's Kappa of 0.67, indicating agreement.

Qualitative comparison

The evaluation will take 20-25 minutes.
The aim of this evaluation, is to rate the images along 4 axis. The detailed criteria of the evaluation as shown below.

IMAGE EVALUATION METRICS
1) ELEMENT INTEGRATION
Score on a scale of 0 to 10, in 1.0 increments, where 10 is the best and 0 is the worst.
Description: How seamlessly different elements are combined within the image.
Criteria:

- Visual Cohesion: Assess whether elements appear as part of a unified scene rather than disjointed parts.
- Object Overlap and Interaction: Check for natural overlaps and interactions between objects, avoiding unnatural placements or intersections.

2) SPATIAL CONSISTENCY
Score on a scale of 0 to 10, in 1 increments, where 10 is the best and 0 is the worst.
Description: Uniformity in style, lighting, and perspective across all elements.
Criteria:

- Stylistic Uniformity: All elements should share a consistent artistic style (e.g., realism, cartoonish).
- Lighting and Shadows: Ensure consistent light sources and shadow directions to maintain realism.
- Perspective Alignment: Elements should adhere to a common perspective, avoiding mismatched viewpoints.

3) SEMANTIC ACCURACY
Score on a scale of 0 to 10, in 1 increments, where 10 is the best and 0 is the worst.
Description: Correct interpretation and representation of each element as described in the prompt.
Criteria:

- Object Accuracy: Objects should match their descriptions in type, attributes, and context.
- Action and Interaction: Actions or interactions between objects should be depicted correctly.

4) AESTHETIC QUALITY
Score on a scale of 0 to 10, in 1 increments, where 10 is the best and 0 is the worst.
Description: Overall visual appeal and artistic quality of the generated image.
Criteria:

- Colour Harmony: Use of colour palettes that are visually pleasing and appropriate for the scene.
- Composition Balance: Balanced arrangement of elements to create an engaging composition.
- Clarity and Sharpness: Images should be clear, with well-defined elements and no unwanted blurriness.

Character 1 Clothing 1

Reference



Method 1



Element Integration

1 2 3 4 5 6 7 8 9 10

Spatial
Consistency

1 2 3 4 5 6 7 8 9 10

Semantic Accuracy

1 2 3 4 5 6 7 8 9 10

Aesthetic Quality

1 2 3 4 5 6 7 8 9 10

Method 2



Element Integration

1 2 3 4 5 6 7 8 9 10

Spatial
Consistency

1 2 3 4 5 6 7 8 9 10

Semantic Accuracy

1 2 3 4 5 6 7 8 9 10

Aesthetic Quality

1 2 3 4 5 6 7 8 9 10

Method 3



Element Integration

1 2 3 4 5 6 7 8 9 10

Spatial
Consistency

1 2 3 4 5 6 7 8 9 10

Semantic Accuracy

1 2 3 4 5 6 7 8 9 10

Aesthetic Quality

1 2 3 4 5 6 7 8 9 10

Method 4



Element Integration

1 2 3 4 5 6 7 8 9 10

Spatial
Consistency

1 2 3 4 5 6 7 8 9 10

Semantic Accuracy

1 2 3 4 5 6 7 8 9 10

Aesthetic Quality

1 2 3 4 5 6 7 8 9 10

Method 5



Element Integration

1 2 3 4 5 6 7 8 9 10

Spatial
Consistency

1 2 3 4 5 6 7 8 9 10

Semantic Accuracy

1 2 3 4 5 6 7 8 9 10

Aesthetic Quality

1 2 3 4 5 6 7 8 9 10

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly outline the train-free framework and its performance, which are heavily supported by experiments in Section 3 and the Appendices.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Limitations and failure modes are explicitly discussed in Appendix F.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The paper relies on empirical observations and theoretical motivation (Appendix B), but does not introduce formal mathematical theorems or proofs.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Implementation details, hyperparameters, and model checkpoints are provided in Section 3.1 and Appendix G.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: Code and data will be open-sourced upon publication; however, detailed implementation instructions are provided within the text to ensure reproducibility.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Hyperparameters, CFG scales, number of denoising steps, and model versions are explicitly detailed in Section 3.1 and Appendix G.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: Standard evaluation benchmarks in this specific sub-field (e.g., GPT-4V pairwise, CLIPScore) generally do not report error bars.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Hardware details (RTX A6000 GPU) are provided in Section 3.1, and an extensive computational cost analysis (MACs and latency) is in Appendix I.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research complies with the Code of Ethics, and an Ethics Statement discussing potential risks and misuse is included in Appendix A.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.

- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Broader impacts, including potential malicious use for deceptive content and reinforcement of biases, are discussed in the Ethics Statement (Appendix A).

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper introduces an inference-time weighting method for existing public models and does not release new foundation models or scraped datasets.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: Direct links and sources for all community LoRAs and base models used in the Flux testbed are provided in Appendix G.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [N/A]

Justification: The paper utilizes existing base models, community LoRAs, and the previously established ComposLoRA testbed framework.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [Yes]

Justification: The full text of instructions and screenshots of the interface used for human expert evaluation are provided in Appendix K.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.

- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The human evaluation involves non-intrusive perceptual quality judgments of AI-generated images by experts, which typically does not meet the threshold for requiring IRB approval.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: GPT-4V is used as an automated evaluator for the generated images. Its specific usage, prompt structure, and code implementation are detailed in Section 3.3 and Appendix H.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.