DuoLoRA : Cycle-consistent and Rank-disentangled Content-Style Personalization

Aniket Roy[1,2]*, Shubhankar Borse[2]†, Shreya Kadambi[2]†, Debasmit Das[2], Shweta Mahajan[2], Risheek Garrepalli[2], Hyojin Park[2], Ankita Nayak[2], Rama Chellappa[1], Munawar Hayat[2], Fatih Porikli[2]
[1]Johns Hopkins University, [2]Qualcomm AI Research‡

Abstract

We tackle the challenge of jointly personalizing content and style from a few examples. A promising approach is to train separate Low-Rank Adapters (LoRA) and merge them effectively, preserving both content and style. Existing methods, such as ZipLoRA, treat content and style as independent entities, merging them by learning masks in LoRA's output dimensions. However, content and style are intertwined, not independent. To address this, we propose DuoLoRA—a content-style personalization framework featuring three key components: (1) rank-dimension mask learning, (2) effective merging via layer priors, and (3) Constyle loss, which leverages cycle-consistency in the merging process. First, we introduce ZipRank, which performs content-style merging within the rank dimension, offering adaptive rank flexibility and significantly reducing the number of learnable parameters. Additionally, we incorporate SDXL layer priors to apply implicit rank constraints informed by each layer's content-style bias and adaptive merger initialization, enhancing the integration of content and style. To further refine the merging process, we introduce Constyle loss, which leverages the cycle-consistency between content and style. Our experimental results demonstrate that DuoLoRA outperforms state-of-the-art content-style merging methods across multiple benchmarks. Project: `https://github.com/aniket004/DuoLoRA.git`

1. Introduction

Text-to-image diffusion models [6, 27, 31] have recently attracted significant interest due to their broad range of applications. These models can produce images with desired content and style, overcoming the limitations of earlier approaches that required extensive training data for neural style transfer [13]. They can generate specific images from just one or two reference images, but blending both content and style remains challenging.

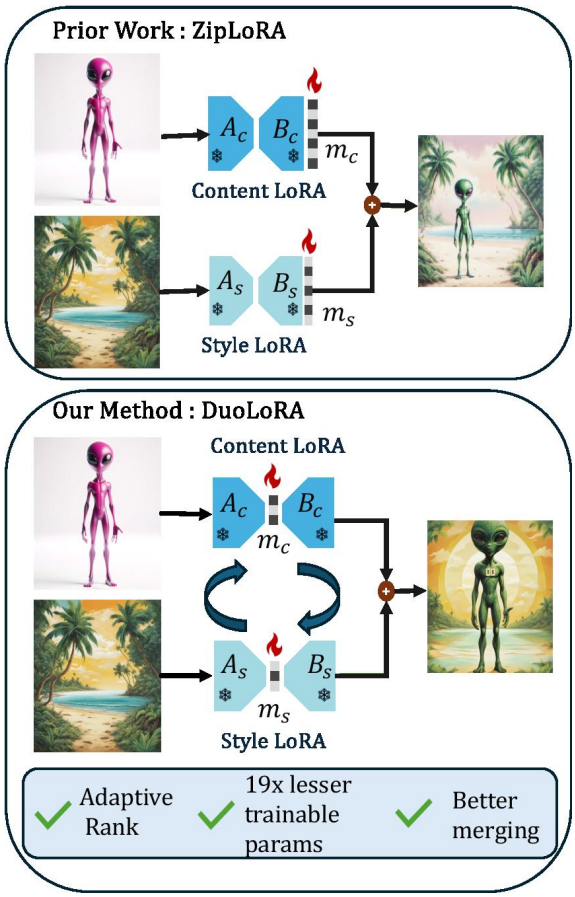


Figure 1. Content and style personalization using DuoLoRA provides (1) adaptive rank flexibility, (2) significantly fewer trainable parameters, and (3) better content-style merging.

Recently, parameter-efficient fine-tuning (PEFT) methods like Low-Rank Adapters (LoRA) [17] have become popular. These methods can capture unique characteristics with a small amount of data. Consequently, content-style personalization can be reframed as a LoRA merging task, where individually trained LoRAs are combined to achieve efficient content-style blending. For example, ZipLoRA [29] merges

*Work done as part of a summer internship.
†Equal contribution
‡Qualcomm AI Research is an initiative of Qualcomm Technologies, Inc.

content and style by learning masks over the LoRA output dimensions, assuming that content and style are independent concepts. This approach provides adequate control over the diffusion model. However, the requirement for fine-tuning during inference is a drawback. Additionally, using the same rank for both content and style LoRAs may not be optimal, as content and style may have different representational requirements depending on each layer.

In this work, we ask: How can we enable adaptive rank flexibility to reduce fine-tuning costs while increasing the separation between content and style distributions? To achieve adaptive rank flexibility across the layers of the diffusion UNet, we analyze the SDXL model architecture [3, 10]. We observe that UNet layers, with smaller resolutions, primarily influence content generation, suggesting that content merges should have higher rank in these layers than style merges. Conversely, the layers, with larger resolutions, are more crucial for style generation, and the rank constraint should be adjusted accordingly [3, 10].

Within this context, we introduce DuoLoRA, a framework for effective content-style merging, composed of three components: (1) ZipRank, which learns masks within the rank dimension, (2) improved merging via layer priors, and (3) Constyle loss, which leverages cycle-consistency across content and style. In ZipRank, we propose learning masks in the rank dimension rather than the output dimension, allowing for adaptive rank adjustment and a substantial reduction in learnable parameters. To facilitate layer-wise rank-adaptive merging, we apply rank constraints during content-style merging, informed by our observations as prior knowledge. This rank constraint is formulated as a nuclear norm minimization problem under a $l1$ sparsity constraint, improving content-style blending with fewer parameters. To further disentangle content and style, we introduce Constyle loss, which leverages cycle-consistency across content and style. This approach, inspired by CycleGAN's [45] treatment of content and style as separate domains optimized through domain translation with minimized reconstruction loss, enables balanced content-style merging. Unlike ZipLoRA, which treats content and style independently, our cycle-consistency loss accounts for their interdependent nature, resulting in improved content-style blending.

In summary, the contributions of this paper are as follows:

- We address content-style personalization as a LoRA merging problem, where we propose learning masks in the rank dimension instead of the output dimension, allowing for adaptive rank flexibility with significantly fewer learnable parameters.
- We analyze the SDXL architecture's layer prior information, finding that layers with lower resolutions primarily contribute to content generation, while layers with higher resolutions focus on style generation. Based on these insights, we introduce explicit rank constraints through nuclear norm minimization under a sparsity constraint to improve merging.
- We introduce Constyle loss, leveraging the cycle-consistency across content and style, and we validate this approach across various benchmarks.

2. Related work

Personalization. Personalizing text-to-image diffusion models has recently attracted significant attention. Early approaches, such as Textual Inversion [11], focus on learning a text token that represents a particular concept, while Dreambooth [27] updates network parameters for personalization. Custom Diffusion [22] makes the process more efficient by fine-tuning only the cross-attention modules. However, these methods typically handle only a single concept or object. In contrast, some approaches aim to personalize both content and style. StyleDrop [31] uses the Muse transformer network to align the style of generated images with a reference image. Other recent style learning methods are - Rb-modulation [26], Instantstyle [36], IP-adapter [42], Magic-insert [28], StyleAlign [16], LoRA-composer [41], Paircustomization [20], [1, 2, 4, 5, 7–9, 14, 17–19, 21, 23–25, 30, 32–40, 43].

LoRA merging. LoRA [17] has proven particularly effective for learning from small datasets. Individual LoRA models are trained for each concept or style, and these concepts can then be combined through LoRA merging. Methods like Concept-Sliders [12] and ControlNet [44] also utilize LoRA merging, though their approaches are primarily suited for text-based editing. Recently, Shah et al. [29] introduced a method for LoRA merging by learning orthogonal masks in the output space of the layer.

3. Method : DuoLoRA

3.1. Problem statement

We address the problem of customizing both content and style from few examples and generate variations conditioned on text prompt. We investigate the problem through the lens of merging individual concepts, where such individual concepts are learned using LoRAs. Therefore we cast the concept-style personalization as a LoRA merging problem (as shown in Fig. 1). However, content and style are not orthogonal, rather those are intertwined concepts. Therefore, entangling such concepts remains challenging.

In order to efficiently merge content and style specific LoRAs, we first learn masks in the rank dimension, with UNet layer-prior informed initialization, explicit rank and sparsity constraint. To further improve performance, we use content-style cycle-consistent merging. We explain this as follows.

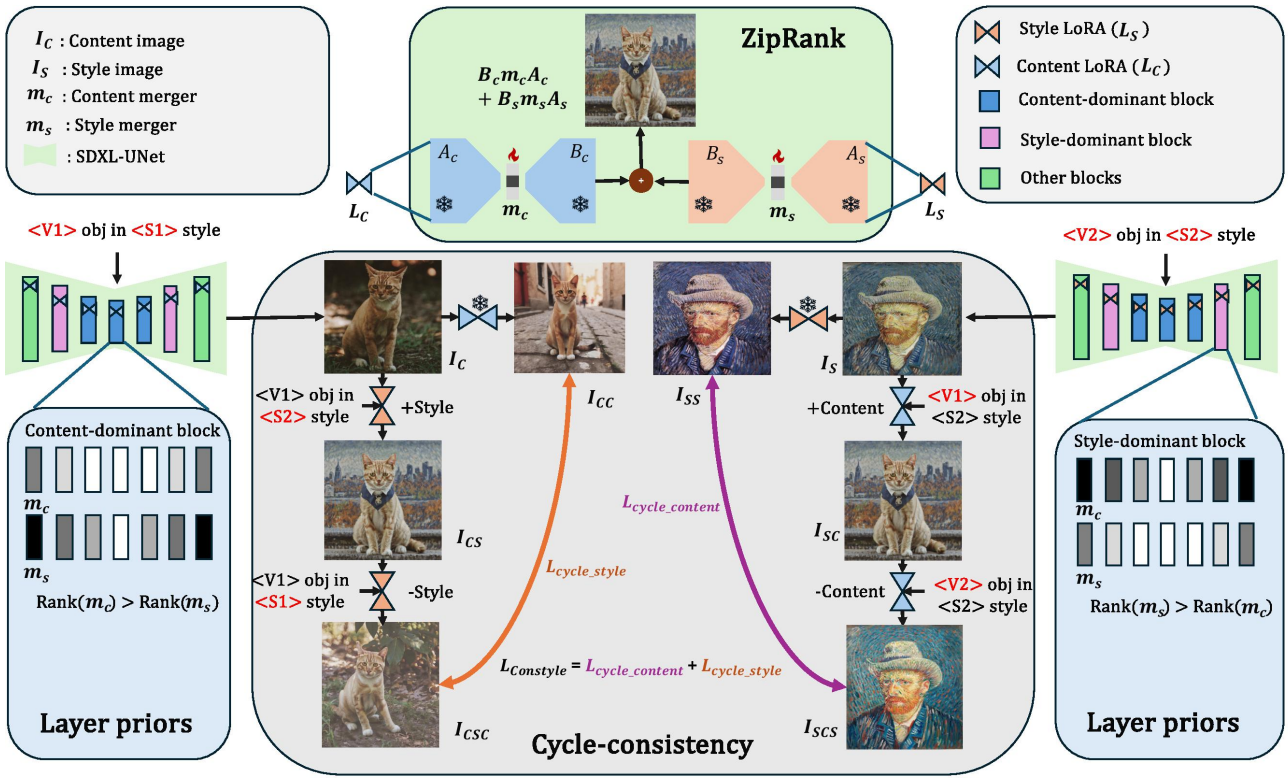


Figure 2. Overview of DuoLoRA. It consists of three components - (1) ZipRank: learning the mask in rank dimension, (2) layer-prior based merging identifying content-dominant and style-dominant blocks of SDXL UNet, (3) cycle-consistency based merging using Constyle loss.

3.2. ZipRank: Merging in Rank space

We start with providing background on LoRA.

Low-Rank Adaptation (LoRA): In LoRA [17], we fine-tune neural networks by approximating weight updates using low-rank matrices. Specifically, the weight update matrix ΔW is parameterized as: $\Delta W = AB = U_r \Sigma_r V_r^\top$, where $A \in \mathbb{R}^{d_{\text{out}} \times r}$, $B \in \mathbb{R}^{r \times d_{\text{in}}}$, and r is the rank of the approximation.

Blending LoRAs optimally presents a significant challenge. A straightforward approach is to use simple arithmetic merging, but this is not efficient. ZipLoRA [29] (Fig. 1) uses the Zipit [34] operation across model weights through learnable masks in the output dimension as defined below, ensuring the masks in the weight space remain orthogonal.

Definition 1. *Output Dimension Masking (ZipLoRA [29]): Here we apply a mask to the output dimension by defining a diagonal mask matrix $M_{out} \in \mathbb{R}^{d_{out} \times d_{out}}$ with entries $n_{ii} \in \{0, 1\}$. The output-masked approximation is:*

$$\Delta W_{out} = M_{out} AB = M_{out} U_r \Sigma_r V_r^\top.$$

Let $d_s = \sum_{i=1}^{d_{out}} n_{ii}$ be the number of active output units.

Our approach, however, focuses on learning masks in the rank dimension (ZipRank, Fig. 2), which greatly reduces the number of learnable parameters and provides adaptive rank flexibility across the LoRA adapters.

Definition 2. *Rank Dimension Masking (ZipRank): We apply a mask to the rank dimension by defining a diagonal mask matrix $M_r \in \mathbb{R}^{r \times r}$ with entries $m_{ii} \in \{0, 1\}$. The rank-masked approximation is:*

$$\Delta W_{rank} = A M_r B = U_r M_r \Sigma_r V_r^\top.$$

Let S denote the set of indices i where $m_{ii} = 1$, and $s = |S|$ is the number of active rank components.

Rank space masking serves two primary motivations: (1) enabling adaptive rank flexibility during LoRA merging and (2) reducing the number of learnable parameters. Unlike ZipLoRA, which targets poorly aligned columns, ZipRank addresses cross-concept interference by modulating the contributions of specific rank components in the latent space, allowing for more nuanced control over concept overlap. In this approach, different concepts (e.g., subject vs. style) are encoded in distinct subspaces within the rank space. Rather than reducing the effective rank, rank-space masking adjusts the relative importance of specific feature interactions, providing fine-grained control without entirely nullifying components. ZipRank is designed to separate the style and content adapters while minimizing the number of learnable

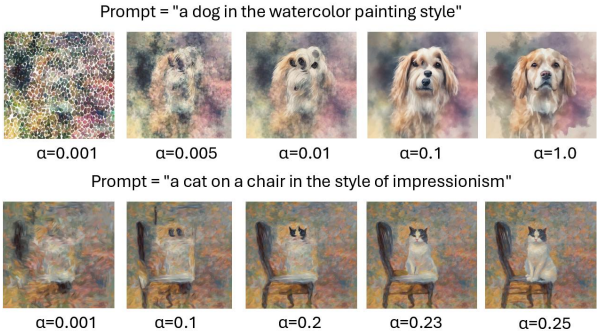


Figure 3. SDXL selective weight scaling. The scaling parameter (α) has been applied to the content-dominant blocks (`up_block.2`, `down_block.2`, and `mid_block`). We observe that with a smaller α, the model fails to generate the content and instead focuses solely on the style. In contrast, increasing α allows the model to generate the content specified in the prompt.

parameters. This approach reduces overfitting, especially when training on small datasets, and introduces adaptive rank flexibility during the merging of LoRAs. We have also shown that under the same parameter budget, the approximation error resulting from rank dimension masking is less than or equal to that from output dimension masking in Theorem. 1 (supplementary).

3.3. Layer priors

3.3.1. How content style is encoded in SDXL

Understanding how content, style, and similar information are encoded within a diffusion UNet is crucial for enabling both local and global edits. We hypothesize that in SDXL, layers with low resolutions (e.g.,`up_block.2`, `down_block.2`, `mid_block`, with resolution < 32) capture localized updates, such as content details, while layers with high resolutions (e.g.,`up_block.1`, `down_block.1`, with resolution >= 32) capture global updates, such as the overall style of the image.

This hypothesis is motivated by the structural design of the SDXL model: layers with low resolutions, typically in the earlier stages of the U-Net, focus on small, localized regions, allowing them to encode essential spatial details, shapes, and structures. This fine-grained attention enables accurate representation of core content elements, like objects and scene layouts, aligning closely with the input prompt.

In contrast, layers with high resolutions, often in deeper sections of the model, capture style by blending and harmonizing details across larger regions. This broader scope makes them well-suited for encoding global stylistic features like color gradients, textures, and lighting. Such attributes require a coherent application across the image rather than precision in spatial detail. Consequently, high resolutions are believed to play a crucial role in generating the overall aesthetic and mood of the image, supporting their role in style encoding.

To verify the hypothesis, we use a layer-freezing simulation via weight scaling. This method allows us to selectively reduce the contributions of specific layers during inference, enabling us to observe the distinct roles that different resolutions play in image generation.

We begin by identifying the layers in the SDXL U-Net architecture associated with low and high resolutions. Layers such as `up_block.2`, `down_block.2`, and `mid_block` have resolutions smaller than 32, while layers like `up_block.1` and `down_block.1` have resolutions larger than 32.

Next, we apply selective weight scaling to simulate a "freezing" effect. We scale the outputs of either low- or high-resolution layers by a small factor, α (e.g., 0.1), to reduce their influence in the generation process. Scaling down the low-resolution layers diminishes their impact on content details, while scaling down the high-resolution layers lessens the effect on style features.

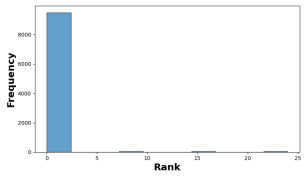
For instance, when using the prompt, `p = "A cat on a chair in the style of impressionism"` during inference, we scale down the weights of the low-resolution layers (`up_block.2`, `down_block.2`, and `mid_block`) by a scalar factor α. By varying α, as shown in Fig. 3, we observe that lower values of α generate the style but fail to capture the object (i.e., the "cat"). As we increase α, the object becomes visible in the generated image. This trend, illustrated in Fig. 3, supports our hypothesis that low-resolution layers (`up_block.2`, `down_block.2`, and `mid_block`) contribute to content generation, while layers with higher resolution (`up_block.1` and `down_block.1`) contribute to style. These observations align with prior findings in the literature [3].

Recent works, like B-LoRA [10], have identified specific layers involved in content (W4 in SDXL) and style (W5 in SDXL). However, these findings are highly specific and may not generalize widely. Rather than adhering strictly to these findings, we incorporate this knowledge in a more generalized way by blending content-style LoRAs. In SDXL, we observe that layers `up_block.2`, `down_block.2`, and `mid_block`, which have lower resolutions ($\leq$ 32), are more involved in content generation. Conversely, layers `up_block.1` and `down_block.1`, with higher resolutions ($\geq$32), contribute more to style generation. Building on this observation, we apply a prior-informed initialization strategy and implicit rank constraint during the merging process. For instance, during merging, we enforce sparsity along with the implicit rank constraint.

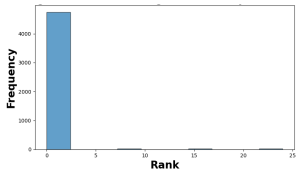
3.3.2. Prior-informed merger initialization

Instead of initializing the merger masks (m_c and m_s in Fig. 2) in the rank dimension with all ones, we incorporate layer-specific information to enhance the merging process. To this end, we use the observation of content-style encoding

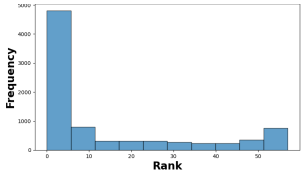
in SDXL architecture made earlier. To capture the dominance of content and style in different layers, the masks for the merged LoRAs are initialized using content (T_{content}) and style (T_{style}) thresholds. In content-dominant layers, more ones are assigned to content merger m_c than style merger m_s, while the reverse is applied in style-dominant layers. The process begins with a normalized random vector, using thresholds to determine priority for content or style based on their ranks, defaulting to all ones when ranks are equal. For other layers, LoRAs are initialized with all ones to adaptively balance content and style merging. As demonstrated in Tab. 8, this adaptive initialization improves performance. The threshold ablation is shown in Tab. 7, and the detailed algorithm is provided in the supplementary.



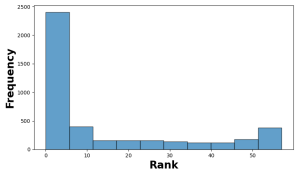
(a) Histogram of rank across low resolution layers in style merger.



(b) Histogram of rank across high resolution layers in content merger.



(c) Histogram of rank across low resolution layers in content merger.



(d) Histogram of rank across high resolution layers in style merger.

Figure 4. Rank analysis. We plot the frequency of ranks across low-resolution layers (i.e., `up_block.2`, `down_block.2`, and `mid_block` with a resolution < 32) and high-resolution layers (resolution >= 32) in content and style mergers post-training. The rank distribution confirms that in SDXL UNet architecture, for low-resolution layers, $Rank(m_c) > Rank(m_s)$, while for high-resolution layers, $Rank(m_s) > Rank(m_c)$.

Algorithm 1: Rank-constrained Layer-prior Merging

Input: SDXL-UNet, content merger (m_c), style merger (m_s)
Output: Loss $\mathcal{L}_{\text{layer\_prior}}$ for merging layers
if *Resolution(SDXL-UNet layer) < 32* **then**
  // $Rank(m_c) > Rank(m_s)$
  $\mathcal{L}_{\text{layer\_prior}} \leftarrow$
  $||m_c||_1 + \lambda\, max\,(0, \|m_s\|_* - \|m_c\|_*)$;
else
  // $Rank(m_s) > Rank(m_c)$
  $\mathcal{L}_{\text{layer\_prior}} \leftarrow$
  $||m_s||_1 + \lambda\, max\,(0, \|m_c\|_* - \|m_s\|_*)$;
end

3.3.3. Prior-informed rank constraint

Based on our observations in Sec. 3.3.1, we propose to use the explicit layer-specific rank constraints during merging. In Fig. 4, we plot the frequency of ranks across low (i.e., `up_block.2`, `down_block.2`, and `mid_block` with < 32 resolution) and high-resolution (>=32 resolution) layers in content and style mergers post training. Through the distribution of the ranks, we can verify that for low resolution layers, $Rank(m_c) > Rank(m_s)$, and for high resolution layers, $Rank(m_s) > Rank(m_c)$ holds in SDXL UNet architecture.

Therefore, for the content-dominant layers, we apply the constraint Rank(m_c) $\geq$ Rank(m_s) during merging, and for the style-dominant layers, we reverse this constraint. In Lemma 1, we demonstrate that this rank constraint simplifies to a nuclear norm minimization problem under a sparsity constraint.

Lemma 1. *Let $m_c \in \mathbb{R}^{m\times n}$ be a matrix representing the content merger and $m_s \in \mathbb{R}^{m\times n}$ be a matrix representing the style merger. The problem of minimizing the l_1-norm of m_c subject to a rank constraint on m_c can be written as:*

$$\min \|m_c\|_1 \quad \textit{subject to} \quad \textit{rank}(m_c) > \textit{rank}(m_s)$$

This problem is non-convex due to the rank constraint. A convex relaxation can be achieved by approximating the rank of a matrix using the nuclear norm $\|\cdot\|_*$, which is the sum of the singular values of the matrix. Thus, the original problem can be relaxed to:

$$\min \|m_c\|_1 \quad \textit{subject to} \quad \|m_c\|_* > \|m_s\|_*$$

where $\|m_c\|_*$ denotes the nuclear norm of m_c, and $\|m_s\|_*$ is the nuclear norm of m_s.

This relaxed problem can be approached via a Lagrangian penalty formulation:

$$\mathcal{L}(m_c, m_s, \lambda) = \|m_c\|_1 + \lambda \max(0, \|m_s\|_* - \|m_c\|_*)$$

for some penalty parameter $\lambda \geq 0$, which enforces the constraint $\|m_c\|_* > \|m_s\|_*$ in the limit as $\lambda \to \infty$.

Proof. Proof is provided in the supplementary material. □

Consequently, in the content-dominant layers (i.e., `up_block.2`, `down_block.2`, and `mid_block`), LoRAs are merged using the layer prior loss $\mathcal{L}_{\text{layer\_prior}}$.

$$\mathcal{L}_{\text{layer\_prior}} = \|m_c\|_1 + \lambda \max(0, \|m_s\|_* - \|m_c\|_*) \quad (1)$$

where m_s and m_c represent the style and content mergers, respectively. Similarly, for the style-dominant layers, the process is reversed, as outlined in Algorithm 1.

3.4. Constyle loss : Cycle-consistent merging

To further improve the content-style alignment, we introduce Constyle loss, leveraging the cycle-consistency between content and style. Inspired by CycleGAN [45], where content and style are transformed across domains with minimized reconstruction loss, we add and then remove style from content, ensuring minimal reconstruction error during LoRA merging. Similarly, content is added and removed from style images, as illustrated in Fig. 2. Both processes provide feedback on the required blending while upholding rank constraints. Next, we describe cycle-consistent merging in detail (color coded for better explanation, refer to Fig. 2).

The cycle-consistent merging is performed as follows:

1. **Generate Individual LoRAs:**

- **Content LoRA:** Given a set of content images I_c, we learn a content LoRA $\mathbf{L_c}$, and tokens **<V1>**, **<S1>** using the prompt $\mathbf{p_c}$ = `"a <V1> object in <S1> style"`.
- **Style LoRA:** For a (small) set of style images I_s, we learn a style LoRA $\mathbf{L_s}$, and tokens **<V2>**, **<S2>** using the prompt $\mathbf{p_s}$ = `"a <V2> object in <S2> style"`.

2. **Cycle-Consistency Across Style:** Starting with a content image I_c, we add noise to create its noisy latent. During the denoising process, the style prompt `"a <V1> object in <S2> style"` is injected via the style LoRA $\mathbf{L_s}$ to produce a style-infused image I_{cs}. This image I_{cs} is then re-noised and denoised with the content prompt $\mathbf{p_c}$ (via $\mathbf{L_s}$) to remove style, resulting in the reconstructed content image I_{csc}. To mitigate mode collapse from limited examples, we also generate a variant I_{cc} by denoising I_c solely with the content LoRA $\mathbf{L_c}$ using $\mathbf{p_c}$. We enforce cycle-consistency in style by minimizing the loss (Fig. 2):

$$\mathcal{L}_{\text{cycle\_style}} = \text{MSE}(I_{cc}, I_{csc})$$

3. **Cycle-Consistency Across Object (Content):** In a similar manner, starting with a style image I_s, we add noise and then denoise using a content prompt `"a <V1> object in <S2> style"` (via the content LoRA $\mathbf{L_c}$) to obtain a content-injected style image I_{sc}. Next, we remove the injected object by denoising I_{sc} with the style prompt $\mathbf{p_s}$ via the content LoRA $\mathbf{L_c}$, yielding I_{scs}. Additionally, we generate a variant I_{ss} by denoising I_s with the style LoRA $\mathbf{L_s}$ using $\mathbf{p_s}$. The cycle-consistency in content is enforced by minimizing (Fig. 2):

$$\mathcal{L}_{\text{cycle\_content}} = \text{MSE}(I_{ss}, I_{scs})$$

The merged LoRA L_m is then trained by minimizing the following loss:

$$\begin{aligned}\mathcal{L}_{\text{constyle}} = &\|(D + L_m)(I_c, \mathbf{p_c}) - (D + \mathbf{L_c})(I_c, \mathbf{p_c})\| \\ &+ \|(D + L_m)(I_s, \mathbf{p_s}) - (D + \mathbf{L_s})(I_s, \mathbf{p_s})\| \\ &+ (\mathcal{L}_{\text{cycle\_style}} + \mathcal{L}_{\text{cycle\_content}})\end{aligned}$$

where D is the text-to-image diffusion model and λ_{cycle} (set to 0.1) controls the weight of the cycle-consistency losses. The overall loss used during merging is given by

$$\mathcal{L} = \lambda_{\text{layer\_prior}}\,\mathcal{L}_{\text{layer\_prior}} + \lambda_{cycle}\,\mathcal{L}_{\text{constyle}}$$

which is further used to train the layer masks. During inference, the merged tokens are provided in the prompt (e.g., `"a <V1> object in <S2> style running"`) to the diffusion model with merged LoRA L_m, generating images that blend the desired content and style. Detailed algorithm is provided in the supplementary.

3.5. Multi-concept stylization

We further extend our approach to handle multi-concept stylization. Given concepts C_1, .. C_n and style S, we first merge individual concept-style LoRAs, $< C_1, S >$,.. $< C_n, S >$, using DuoLoRA/baselines and then combine them via naive merging: $C_{1,2,..,n,S} = \alpha_1 < C_1, S > +.. + \alpha_n < C_n, S >$, where $\alpha_i = 1/n$. During inference, we use directional prompting (i.e., object in left/right etc) with the merged LoRA $C_{1,2,..,n,S}$, using prompt `p = "a` $< C1 >$ `object on the left and a` $< C2 >$ `object on the right .. in` $< S >$ `style"`. In this way, we extend DuoLoRA for multiple concepts.

4. Experiments

Dataset. We experiment on four datasets (datasets and splits provided in supplementary).
(1) Dreambooth-StyleDrop: We choose diverse set of content images from the Dreambooth dataset [27], which consists of images of 30 subjects with 4-5 images per subjects. Style images are chosen from StyleDrop [31] dataset, where a single image is used per style.
(2) Subjectplop: subjectplop [28] contains a single image for both content and style.
(3) Subjectplop-StyleDrop: We also benchmark on cross dataset, i.e., contents are taken from subjectplop dataset [28], and style images are taken from StyleDrop [31] dataset. Note here also, the content and style contains a single image.
(4) Custom101-StyleDrop: We conduct experiments on the real-world object-centric Custom101 dataset [22] (with 101 objects like human faces and everyday items) using styles from the Styledrop dataset [31].
Metrics. For content similarity, we use DINO similarity score [27], i.e., the average pairwise cosine similarity of DINO ViT-B/6 embeddings of the content and generated images. For style similarity, we use the CLIP-I metric [27], which is the average pairwise cosine similarity between CLIP embeddings of the style and generated images. Text similarity (CLIP-T [27]) is computed as the average cosine similarity between CLIP's text and prompt embeddings. We also report the Contrastive Style Descriptor-style (CSD-s) metric [32], which is more appropriate to evaluate style similarity.

Table 1. Performance comparison of content and style merging across different datasets and methods

Method	Dreambooth + StyleDrop				Subjectplop				Subjectplop + StyleDrop				Custom101 + StyleDrop				# Params (M)	Param storage (MB)	Training time (m)
	DINO	CLIP-I	CLIP-T	CSD-s	DINO	CLIP-I	CLIP-T	CSD-s	DINO	CLIP-I	CLIP-T	CSD-s	DINO	CLIP-I	CLIP-T	CSD-s			
Naïve Merging	0.47	0.64	0.266	0.44	0.48	0.59	0.263	0.30	0.42	0.49	0.274	0.12	0.40	0.39	0.204	0.18	-	-	-
B-LoRA [10] (ECCV'24)	0.45	0.57	0.281	0.28	0.64	0.57	0.275	0.32	0.63	0.56	0.281	0.14	0.49	0.51	0.263	0.25	-	-	-
ZipLoRA [29] (ECCV'24)	0.53	0.65	0.285	0.41	0.75	0.62	0.288	0.35	0.87	0.56	0.289	0.16	0.54	0.58	0.286	0.30	1.33	6.5	5.48
ZipRank	0.53	0.64	0.287	0.42	0.71	0.62	0.291	0.35	0.86	0.56	0.296	0.17	0.56	0.58	0.295	0.32	0.07	0.35	**5.28**
ZipRank + Layer-Priors	0.54	0.67	0.293	0.45	0.73	0.63	0.310	0.37	0.90	0.56	0.302	0.18	0.59	0.60	0.307	0.35	0.07	0.35	6.02
DuoLoRA	**0.56**	**0.69**	**0.314**	**0.48**	**0.78**	**0.65**	**0.318**	**0.40**	**0.90**	**0.58**	**0.319**	**0.20**	**0.61**	**0.62**	**0.316**	**0.37**	**0.07**	**0.35**	6.38

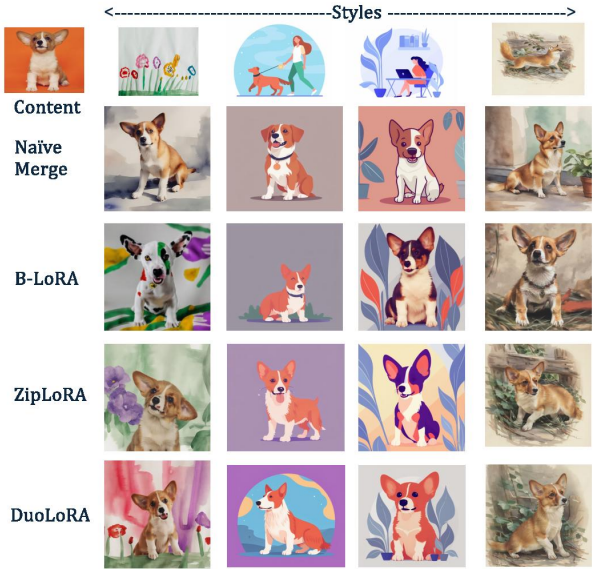


Figure 5. Qualitative Results on Dreambooth + StyleDrop.

Implementation details. In all our experiments, we have used SDXL v1.0 as our base model. To train the content and style LoRAs, we use Dreambooth finetuning with LoRA of rank 64. We update the LoRA weights using Adam optimizer for 1000 steps with batch size of 1 and learning rate of 5e-4. The text encoder of SDXL remains frozen during the LoRA finetuning.

During the merging, we train the mergers in the rank dimension for 100 steps, with both the layer-prior loss ($\mathcal{L}_{layer\_prior}$) and the cycle loss ($\mathcal{L}_{constyle}$), with Adam optimizer and a learning rate of 0.01. The hyperparameters are choosen as follows, $\lambda_{cycle} = 0.01$ and $\lambda_{layer\_prior} = 0.1$.

Baselines. For personalized stylization, we compare our method with the following baselines. (1) Naive merging: The content and style adapters are added together with a blending scalar parameter during inference. This merging is training free. (2) B-LoRA [10]: Specific blocks are learnt for content and style in SDXL. (3) ZipLoRA [29]: A weighting vector is learned at the output dimension of each adapter, such that their similarity is reduced. We used the parameters from ZipLoRA paper [29]. and (4) Paircustomization [20].

Quantitative and Qualitative Results. We compare our method with Naive merging, B-LoRA and ziplora baselines. Tab. 1 shows the comparison with different datasets and baseline methods. Our method outperforms the SOTA in all the DINO, CLIP and CSD metrics. Qualitative results are shown in Fig. 5, Fig. 6 and Fig. 7. Visually it is also evident that our method outperform the baselines. We also compare

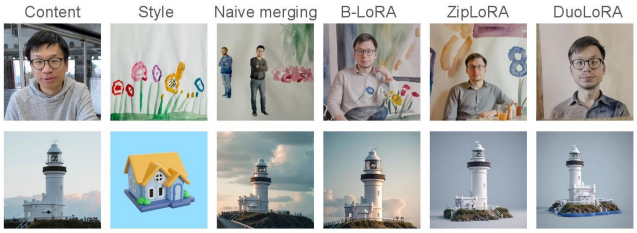


Figure 6. Qualitative Results on Custom101 (best viewed in color).

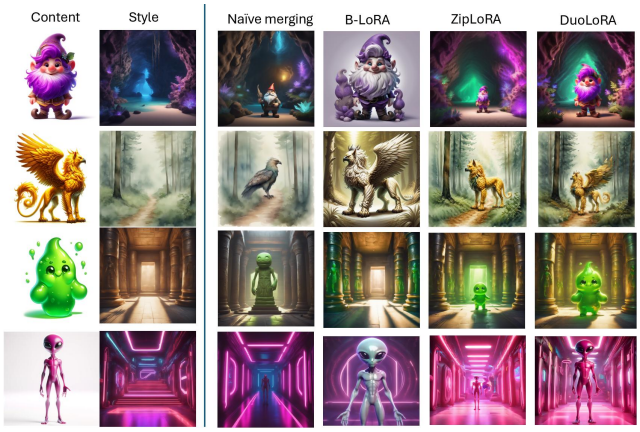


Figure 7. Qualitative Results on Subjectplop (best viewed in color).

Figure 8. Recontextualization through prompts.

DuoLoRA with Pair-customization [20] in Tab. 2, using their test set for a fair evaluation since their approach requires paired object and stylized images (details and qualitative results in supplementary). To demonstrate that DuoLoRA generated images are not biased toward the content of the style images, we measured the DINO similarity between the content, style, and generated images. On the Dreambooth-StyleDrop benchmark, the average DINO similarity is 0.56 w.r.t content images and 0.25 w.r.t style images, indicating minimal content bias from the style images.

Recontextulization. Tab. 1 and Fig. 8 show DuoLoRA successfully recontextualizes through text prompts, measured by CLIP-T, (e.g., "a <V> object in <S> style riding a bicycle") while seamlessly blending content and style (more in supplementary). Fig. 8 also demonstrates that our blending preserves the model's finer text-based editing capabilities,

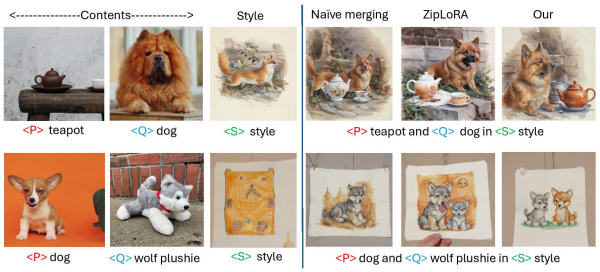


Figure 9. Multi-concept stylization comparison (2 concepts).

such as depicting "sleeping" through closed eyes.

Table 2. Comparison with [20]

Method	DINO	CLIP-I	CSD-s
Paircustomization [20]	0.56	0.65	0.47
DuoLoRA	0.62	0.69	0.50

Table 3. Same #params.

Method	DINO	CLIP-I	CSD-s	#Params
ZipLoRA	0.53	0.65	0.41	1.33M
DuoLoRA	0.57	0.73	0.50	1.33M

Multi-concept stylization. The multi-concept stylization results are visualized in Fig. 9,10, where our method shows qualitative improvements over the baselines in preserving both content and style details. For instance, as shown in Fig. 9, naive merging often loses details of either content, style, or both. Similarly, ZipLoRA struggles to retain content details, while our method successfully captures and maintains both content and style elements. We combine 2, 3, 4 concepts from Dreambooth dataset and styles from StyleDrop dataset and the results are shown in Tab. 4, Fig. 9 and Fig. 10. Details and more results are in the supplementary.

Other diffusion models. We evaluate DuoLoRA on the SSD-1B [15] and Segmind-Vega [15] architectures, demonstrating its generalizability across architectures (Tab. 5).

Runtime and storage. We compare runtime and extra storage required for the mergers in DuoLoRA vs baselines in Tab. 1 on an NVIDIA A6000 (24GB RAM). Our method slightly increase training time while reducing extra storage requirements compared to the baselines.

User study. Since perceptual metrics are not always reliable, we also conducted a human preference study using Amazon Mechanical Turk (AMT) for assesing the content-style alignment. We asked 50 unbiased users rank our method against baselines (i.e., "Naive merging", "B-LoRA", "ZipLoRA", "DuoLoRA", "None is satisfactory"), totaling 1000 questionnaires. The aggregate responses in Tab. 6 show that DuoLoRA generated images significantly outperformed the baselines by a large margin (50%). Further details are provided in the Supplementary.

Ablations. We perform ablations on the loss and initialization when applied to content (Dreambooth dataset) and style (StyleDrop dataset) in Tab. 8. Ablation of initialization parameters ($T_{content}$ and T_{style}) is also provided in Tab. 7. We also present a hyperparameter sensitivity analysis on Dreambooth-StyleDrop in Tab. 9 w.r.t loss coefficients ($\lambda_{layer\_prior}, \lambda_{cycle}$). With same number of parameters compared to baseline (ZipLoRA), DuoLoRA obtains better performance gains in Tab. 3. More ablation results are provided in the supplementary material.

Limitations. At present, our approach handles only two concepts simultaneously, limiting its applicability for merging multiple concepts jointly. We aim to address this limitation in future work.

Table 4. Multi-concept comparison

Method	2-concepts			3-concepts			4-concepts		
	DINO	CLIP-I	CSD-s	DINO	CLIP-I	CSD-s	DINO	CLIP-I	CSD-s
Naïve Merging	0.38	0.63	0.35	0.35	0.56	0.30	0.28	0.55	0.25
ZipLoRA	0.40	0.64	0.42	0.38	0.60	0.34	0.29	0.58	0.31
DuoLoRA	0.45	0.66	0.47	0.40	0.64	0.39	0.32	0.63	0.35

Table 5. Comparison w.r.t architectures

Method	SSD-1B				Segmind-Vega			
	DINO	CLIP-I	CSD-s	#params(M)	DINO	CLIP-I	CSD-s	#params(M)
Naïve Merging	0.35	0.55	0.38	-	0.33	0.57	0.42	-
ZipLoRA	0.42	0.65	0.46	0.61	0.40	0.67	0.50	0.26
DuoLoRA	0.44	0.71	0.52	0.03	0.41	0.72	0.56	0.02

Figure 10. Multi-concept stylization (4 concepts).

Table 6. User study

None	Naive merging	B-LoRA	ZipLoRA	DuoLoRA
0.0%	10%	18%	22%	50%

Table 7. Ablation of content and style threshold ($T_{content}, T_{style}$)

Method	λ_{layer_prior}	$T_{content}$	T_{style}	DINO	CLIP-I	CSD-s
ZipRank	0.1	-	-	0.53	0.64	0.42
ZipRank + Init	0.1	0.1	0.0	0.56	0.65	0.43
ZipRank + Init	0	0.75	0.50	0.57	0.64	0.41
ZipRank + Init	0	1	0.75	0.53	0.64	0.42
ZipRank + Init	0.1	0.75	0.50	0.50	0.64	0.44
ZipRank + Init	0.1	1	0.75	0.47	0.64	0.43

Table 8. Ablations of components

ZipRank	Merger Initialization	Nuclear Norm loss	Sparsity loss	Cycle Loss (Style)	Cycle Loss (Content+Style)	DINO	CLIP-I	CSD-s
✓	✗	✗	✗	✗	✗	0.530	0.647	0.424
✓	✓	✗	✗	✗	✗	0.560	0.652	0.435
✓	✓	✓	✗	✗	✗	0.577	0.637	0.400
✓	✓	✓	✗	✗	✗	0.516	0.679	0.476
✓	✓	✓	✓	✗	✗	0.542	0.670	0.458
✓	✓	✓	✓	✓	✗	0.507	0.704	0.519
✓	✓	✓	✓	✓	✓	0.560	0.693	0.482

Table 9. Ablation of parameters

Parameters ($\lambda_{layer\_prior}/\lambda_{cycle}$)	0/0	0/1	0/0.01	1/0	0.1/0	0.1/0.1	0.01/0.01	0.1/0.01	1.0/0.01
DINO	0.53	0.45	0.47	0.57	0.55	0.52	0.48	0.51	0.56
CLIP-I	0.64	0.70	0.68	0.63	0.64	0.64	0.65	0.66	0.69
CSD-s	0.42	0.52	0.48	0.40	0.42	0.43	0.45	0.46	0.48

5. Conclusion

We investigate content and style personalization via LoRA merging, introducing DuoLoRA—a content-style personalization framework with three core components: (1) learning a mask in the rank dimension, (2) merging informed by layer priors, and (3) Constyle loss that leverages cycle-consistency between content and style. By learning the mask in the rank dimension, DuoLoRA enables adaptive rank flexibility with a significant reduction in trainable parameters (19x fewer). To further refine the merging process, we apply explicit rank constraints informed by layer priors and adaptive initialization. Additionally, we introduce Constyle loss, which uses content-style cycle-consistency to enhance merging. Experiments across multiple benchmarks show that our approach outperforms state-of-the-art methods.

6. Acknowledgement

A. Roy and R. Chellappa acknowledge support through a fellowship from JHU + Amazon Initiative for Interactive AI (AI2AI) and ONR MURI grant N00014-20-1-2787.

References

[1] Aishwarya Agarwal, Srikrishna Karanam, and Balaji Vasan Srinivasan. Training-free color-style disentanglement for constrained text-to-image synthesis. *arXiv preprint arXiv:2409.02429*, 2024.

[2] Omri Avrahami, Rinon Gal, Gal Chechik, Ohad Fried, Dani Lischinski, Arash Vahdat, and Weili Nie. Diffuhaul: A training-free method for object dragging in images. In *SIGGRAPH Asia 2024 Conference Papers*, pages 1–12, 2024.

[3] Samyadeep Basu, Keivan Rezaei, Priyatham Kattakinda, Vlad I Morariu, Nanxuan Zhao, Ryan A Rossi, Varun Manjunatha, and Soheil Feizi. On mechanistic knowledge localization in text-to-image generative models. In *Forty-first International Conference on Machine Learning*, 2024.

[4] Xiuli Bi, Jian Lu, Bo Liu, Xiaodong Cun, Yong Zhang, Weisheng Li, and Bin Xiao. Customttt: Motion and appearance customized video generation via test-time training. *arXiv preprint arXiv:2412.15646*, 2024.

[5] Shubhankar Borse, Kartikeya Bhardwaj, Mohammad Reza Karimi Dastjerdi, Hyojin Park, Shreya Kadambi, Shobitha Shivakumar, Prathamesh Mandke, Ankita Nayak, Harris Teague, Munawar Hayat, et al. Subzero: Composing subject, style, and action via zero-shot personalization. *arXiv preprint arXiv:2502.19673*, 2025.

[6] Huiwen Chang, Han Zhang, Jarred Barber, AJ Maschinot, Jose Lezama, Lu Jiang, Ming-Hsuan Yang, Kevin Murphy, William T Freeman, Michael Rubinstein, et al. Muse: Text-to-image generation via masked generative transformers. *arXiv preprint arXiv:2301.00704*, 2023.

[7] Jooyoung Choi, Chaehun Shin, Yeongtak Oh, Heeseung Kim, and Sungroh Yoon. Style-friendly snr sampler for style-driven generation. *arXiv preprint arXiv:2411.14793*, 2024.

[8] Nadav Z Cohen, Oron Nir, and Ariel Shamir. Conditional balance: Improving multi-conditioning trade-offs in image generation. *arXiv preprint arXiv:2412.19853*, 2024.

[9] Zhaoli Deng, Kaibin Zhou, Fanyi Wang, and Zhenpeng Mi. Magicstyle: Portrait stylization based on reference image. *arXiv preprint arXiv:2409.08156*, 2024.

[10] Yarden Frenkel, Yael Vinker, Ariel Shamir, and Daniel Cohen-Or. Implicit style-content separation using b-lora. *arXiv preprint arXiv:2403.14572*, 2024.

[11] Rinon Gal, Yuval Alaluf, Yuval Atzmon, Or Patashnik, Amit H Bermano, Gal Chechik, and Daniel Cohen-Or. An image is worth one word: Personalizing text-to-image generation using textual inversion. *arXiv preprint arXiv:2208.01618*, 2022.

[12] Rohit Gandikota, Joanna Materzynska, Tingrui Zhou, Antonio Torralba, and David Bau. Concept sliders: Lora adaptors for precise control in diffusion models. *arXiv preprint arXiv:2311.12092*, 2023.

[13] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2414–2423, 2016.

[14] Yanqi Ge, Jiaqi Liu, Qingnan Fan, Xi Jiang, Ye Huang, Shuai Qin, Hong Gu, Wen Li, and Lixin Duan. Tuning-free adaptive style incorporation for structure-consistent text-driven style transfer. *arXiv preprint arXiv:2404.06835*, 2024.

[15] Yatharth Gupta, Vishnu V Jaddipal, Harish Prabhala, Sayak Paul, and Patrick Von Platen. Progressive knowledge distillation of stable diffusion xl using layer level loss. *arXiv preprint arXiv:2401.02677*, 2024.

[16] Amir Hertz, Andrey Voynov, Shlomi Fruchter, and Daniel Cohen-Or. Style aligned image generation via shared attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4775–4785, 2024.

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

[18] Juncheng Hu, Ximing Xing, Jing Zhang, and Qian Yu. Vectorpainter: Advanced stylized vector graphics synthesis using stroke-style priors. *arXiv preprint arXiv:2405.02962*, 2024.

[19] Ruixiang Jiang and Changwen Chen. Artist: Aesthetically controllable text-driven stylization without training. *arXiv preprint arXiv:2407.15842*, 2024.

[20] Maxwell Jones, Sheng-Yu Wang, Nupur Kumari, David Bau, and Jun-Yan Zhu. Customizing text-to-image models with a single image pair. *arXiv preprint arXiv:2405.01536*, 2024.

[21] Hubert Kompanowski and Binh-Son Hua. Dream-in-style: Text-to-3d generation using stylized score distillation. *arXiv preprint arXiv:2406.18581*, 2024.

[22] 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.

[23] Chang Liu, Viraj Shah, Aiyu Cui, and Svetlana Lazebnik. Unziplora: Separating content and style from a single image. *arXiv preprint arXiv:2412.04465*, 2024.

[24] Tillmann Ohm, Andres Karjus, Mikhail V Tamm, and Maximilian Schich. fruit-salad: A style aligned artwork dataset to reveal similarity perception in image embeddings. *Scientific Data*, 12(1):254, 2025.

[25] 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.

[26] Litu Rout, Yujia Chen, Nataniel Ruiz, Abhishek Kumar, Constantine Caramanis, Sanjay Shakkottai, and Wen-Sheng Chu. Rb-modulation: Training-free personalization of diffusion models using stochastic optimal control. *arXiv preprint arXiv:2405.17401*, 2024.

[27] 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.

[28] Nataniel Ruiz, Yuanzhen Li, Neal Wadhwa, Yael Pritch, Michael Rubinstein, David E Jacobs, and Shlomi Fruchter. Magic insert: Style-aware drag-and-drop. *arXiv preprint arXiv:2407.02489*, 2024.

[29] 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, 2025.

[30] 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. *arXiv preprint arXiv:2412.05148*, 2024.

[31] 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. *arXiv preprint arXiv:2306.00983*, 2023.

[32] Gowthami Somepalli, Anubhav Gupta, Kamal Gupta, Shramay Palta, Micah Goldblum, Jonas Geiping, Abhinav Shrivastava, and Tom Goldstein. Measuring style similarity in diffusion models. *arXiv preprint arXiv:2404.01292*, 2024.

[33] Bingjie Song, Xin Huang, Ruting Xie, Xue Wang, and Qing Wang. Style3d: Attention-guided multi-view style transfer for 3d object generation. *arXiv preprint arXiv:2412.03571*, 2024.

[34] George Stoica, Daniel Bolya, Jakob Bjorner, Pratik Ramesh, Taylor Hearn, and Judy Hoffman. Zipit! merging models from different tasks without training. *arXiv preprint arXiv:2305.03053*, 2023.

[35] XINYUE SUN, Jing Guo, Shuai Yang, Kai Wang, et al. Sast: Semantic-aware stylized text-to-image generation. *Jing and Yang, Shuai and Wang, Kai, Sast: Semantic-Aware Stylized Text-to-Image Generation*.

[36] Haofan Wang, Matteo Spinelli, Qixun Wang, Xu Bai, Zekui Qin, and Anthony Chen. Instantstyle: Free lunch towards style-preserving in text-to-image generation. *arXiv preprint arXiv:2404.02733*, 2024.

[37] Zhiyu Xie, Yuqing Zhang, Xiangjun Tang, Yiqian Wu, Dehan Chen, Gongsheng Li, and Xiaogang Jin. Styletex: Style image-guided texture generation for 3d models. *ACM Transactions on Graphics (TOG)*, 43(6):1–14, 2024.

[38] Peng Xing, Haofan Wang, Yanpeng Sun, Qixun Wang, Xu Bai, Hao Ai, Renyuan Huang, and Zechao Li. Csgo: Content-style composition in text-to-image generation. *arXiv preprint arXiv:2408.16766*, 2024.

[39] Yu Xu, Fan Tang, Juan Cao, Yuxin Zhang, Oliver Deussen, Weiming Dong, Jintao Li, and Tong-Yee Lee. Break-for-make: Modular low-rank adaptations for composable content-style customization. *arXiv preprint arXiv:2403.19456*, 2024.

[40] Youcan Xu, Zhen Wang, Jun Xiao, Wei Liu, and Long Chen. Freetuner: Any subject in any style with training-free diffusion. *arXiv preprint arXiv:2405.14201*, 2024.

[41] 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.

[42] Hu Ye, Jun Zhang, Sibo Liu, Xiao Han, and Wei Yang. Ip-adapter: Text compatible image prompt adapter for text-to-image diffusion models. *arXiv preprint arXiv:2308.06721*, 2023.

[43] Gong Zhang, Kihyuk Sohn, Meera Hahn, Humphrey Shi, and Irfan Essa. Finestyle: Fine-grained controllable style personalization for text-to-image models. *Advances in Neural Information Processing Systems*, 37:52937–52961, 2024.

[44] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3836–3847, 2023.

[45] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.