
Personalized Image Editing in Text-to-Image Diffusion Models via Collaborative Direct Preference Optimization

Connor Dunlop Matthew Zheng* Kavana Venkatesh* Pinar Yanardag
Virginia Tech, Blacksburg, VA
{cdunlop, pinary}@vt.edu
<http://personalized-editing.github.io>



Figure 1: Our framework performs *personalized* image editing aligned with user’s preference via a novel DPO objective that learns user preferences while leveraging collaborative signals from like-minded individuals.

Abstract

Text-to-image (T2I) diffusion models have made remarkable strides in generating and editing high-fidelity images from text. Yet, these models remain fundamentally generic, failing to adapt to the nuanced aesthetic preferences of individual users. In this work, we present the first framework for personalized image editing in diffusion models, introducing Collaborative Direct Preference Optimization (C-DPO), a novel method that aligns image edits with user-specific preferences while leveraging collaborative signals from like-minded individuals. Our approach encodes each user as a node in a dynamic preference graph and learns embeddings via a lightweight graph neural network, enabling information sharing across users with overlapping visual tastes. We enhance a diffusion model’s editing capabilities by integrating these personalized embeddings into a novel DPO objective, which jointly optimizes for individual alignment and neighborhood coherence. Comprehensive experiments, including user studies and quantitative benchmarks, demonstrate that our method consistently outperforms baselines in generating edits that are aligned with user preferences.

*Equal contribution.

1 Introduction

Text-to-image (T2I) diffusion models have achieved remarkable success in visual content generation, enabling high-quality image synthesis from textual descriptions [32, 31, 18]. Building on this success, recent advancements have begun to repurpose diffusion models for image editing – allowing users to modify a given image via text prompts or other guidance [42, 5]. Despite progress, achieving a desired edit often demands substantial manual effort. Existing editing approaches do not account for the individual user’s preferences. They treat image editing as a one-size-fits-all problem: the model does not adapt to a particular user’s style, and it assumes the same definition of a ‘good’ edit for everyone.

In other domains like natural language processing, it is well recognized that different users have unique tastes and requirements, and models can be adapted accordingly [38, 30]. Analogously, in image editing each user may exhibit distinct preferences - one user might favor images with brighter, more saturated colors, while another prefers muted tones and centered composition. Current text-to-image models and editors ignore these nuances, effectively optimizing for an average preference that may not align with any particular user. This lack of personalization means users must continuously correct or fine-tune outputs that do not suit their aesthetic, which limits the practicality of these AI editors in real creative workflows.

In this work, we take the first step towards *personalized image editing* by proposing a framework that learns and adapts to an individual user’s editing preferences. Our method models how each user prefers their images to be modified, capturing aspects such as stylistic choices, color palettes, object attributes, and other consistently favored visual traits. By learning from user-specific preferences, our framework aligns image edits with the user’s intent, producing results that better reflect their personal aesthetic. Importantly, we also recognize that collaboration across users can amplify personalization. Users often fall into clusters with overlapping tastes; by sharing preference information among like-minded users, we can generalize better from limited data. Consider a home-decor hobbyist who frequently edits interior photos to give them a rustic living-room feel. They always add a stone fireplace and distressed-leather sofa, but they have never thought to include exposed wooden ceiling beams. Several like-minded users in our preference graph routinely pair the fireplace-and-sofa combo with rough-hewn ceiling beams to complete the rustic look. Even though the target user has never requested the beams explicitly, our collaborative mechanism learns this object-level association from neighboring users’ histories and can automatically insert the beams in future edits. The result is an image that remains true to the user’s rustic touches while enriching the scene with an additional detail they are likely to appreciate.

In this paper, we propose a novel Direct Preference Optimization (DPO) framework for collaborative personalized editing. Specifically, we introduce a DPO-based learning approach that aligns a diffusion model’s image edits with individual user preferences while simultaneously leveraging feedback from a community of users. To enable this capability, we propose a novel synthetic dataset of user editing preferences; a rich corpus of editing examples with annotations linking them to specific users and their satisfaction. We represent the users in this dataset as nodes in a graph, which captures similarities in their editing behavior. This graph-based collaborative learning, together with the DPO objective, allows our model to learn a wide spectrum of personalized editing styles within a single unified model. In summary, our contributions are three-fold:

- To the best of our knowledge, we introduce the first formulation of personalized text-to-image editing. We define a new problem setting where an editing model is tailored to an individual user’s preferences, moving beyond the one-size-fits-all paradigm in image editing.
- We propose a novel training framework called, Collaborative Direct Preference Optimization, introducing a graph-structured regularization term into the DPO loss, explicitly modeling and leveraging collaborative relationships among user embeddings. This structured collaboration allows the model to capture nuanced preferences implicitly shared among like-minded users.
- We curate a novel synthetic dataset comprising 144K editing preferences which includes annotated examples of edits grouped by user. This dataset provides a valuable benchmark for studying personalization in image editing and facilitates quantitative evaluation of models in this setting.
- Our source code and dataset are publicly available at <http://personalized-editing.github.io>.

2 Related Work

Text-to-Image Editing Text-to-image diffusion models [36, 14, 34, 32] are frequently used for both image generation and editing because of their high-fidelity generation capabilities. Instruct Pix2Pix [5] enables flexibility across edit types with user prompts without requiring task-specific supervision, however, its performance can degrade for highly detailed or spatially complex edits. Blended Latent Diffusion [3] and SDEdit [23] provide more fine-grained control with diffusion models whereas ControlNet [42] conditions the model on additional structural inputs to maintain the spatial consistency while also allowing for localized edits.

User Preference Optimization Reinforcement Learning from Human Feedback (RLHF) [26] has been used to optimize Large Language Models (LLMs) [6, 29, 37] and multimodal LLMs [2, 22, 8, 1] to better align with human preferences. It involves training a reward model based on pairs of preferred and rejected data to guide the fine-tuning of the language model. A simpler alternative to RLHF is Direct Preference Optimization (DPO) [30], which directly optimizes the model parameters on preference data rather than the need for a separate reward model or reinforcement learning loop. The applications of DPO have extended beyond LLMs to vision domains. Adaptations such as Diffusion-DPO [38], have been proposed to similarly align diffusion models to generate images in accordance with human preferences. DPO has been used to further align models with individual users rather than an entire population [20].

Personalized Image Generation The expressive capabilities of T2I models have spurred research in user personalization. Recent methods have aimed to fit the preferences of an individual user. ViPER [35] fine-tunes a VLM on a dataset of AI-generated user comments on images to extract information on user preferences. It then includes these preferences in the prompt and classifier-free-guidance scale to align the model with the user during generation. However, the dependence on detailed comments provided by the user limits its ability to tailor specific image features in the absence of high-quality comments. Other methods have employed reward optimization to learn to represent user preferences in generation. PASTA [24] employs a reinforcement learning agent to iteratively refine a text prompt based on sequential user interactions before feeding to a T2I model. PPD [7] uses a VLM to compare liked and disliked images to train additional cross-attention layers using a Diffusion-DPO reward objective. Pigeon [40] trains a mask generator for user history and feeds to a multimodal LLM trained with a DPO objective to generate visual tokens used to guide the diffusion model. Although these methods aim to learn user preferences across individual history, they fail to consider that some user preferences may not exist within a user’s own history, but can be inferred from that of similar or relevant users. Furthermore, none of these methods have addressed the task of user-personalized image *editing*, a much more complex task that requires maintaining consistency across edits while also capturing the nuanced editing preferences of users.

3 Background

The alignment of large language and diffusion models is often framed as a problem of *preference learning* in which a model is encouraged to rank preferred outputs higher than rejected ones. Traditional pipelines first train a separate *reward model* from pairwise human-preference data and then maximize the expected reward with reinforcement learning (RLHF) [26]. Although effective, RLHF adds algorithmic complexity, instability, and significant compute overhead.

Direct Preference Optimization. [30] recently introduced **Direct Preference Optimization** (DPO), a lightweight alternative that *removes* the explicit reward model and RL stage. Let $\mathcal{D} = \{(x_i, y_i^+, y_i^-)\}_{i=1}^N$ be a dataset of prompts x_i paired with a **chosen** response y_i^+ and a **rejected** response y_i^- . Denote by $\pi_\theta(\cdot | x)$ the conditional distribution of the *trainable* policy and by $\pi_{\text{ref}}(\cdot | x)$ a frozen *reference* policy (e.g. the pre-SFT checkpoint). DPO maximizes the log-odds that the trainable policy assigns higher probability to y^+ than to y^- , while staying close to the reference. The resulting objective is a simple logistic loss

$$\mathcal{L}_{\text{DPO}}(\theta) = -\mathbb{E}_{(x, y^+, y^-) \sim \mathcal{D}} \left[\log \sigma(\beta [\Delta_\theta(x, y^+, y^-) - \Delta_{\text{ref}}(x, y^+, y^-)]) \right], \quad (1)$$

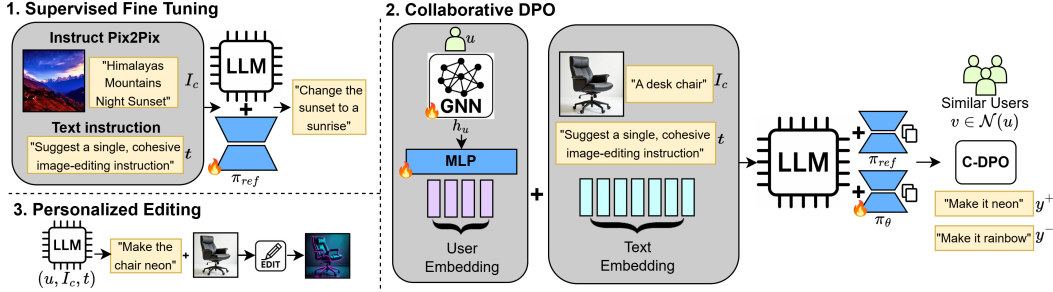


Figure 2: 1) We first fine-tune a language model so it can generate precise editing instructions. 2) We then introduce a graph-aware DPO objective that leverages collaborative user data to learn individual editing preferences. 3) After training, the system takes an input image and a user profile, produces tailored editing instructions, and outputs the corresponding personalized edit.

where $\sigma(\cdot)$ is the sigmoid, $\beta > 0$ is a temperature controlling conservatism, and Δ_θ and Δ_{ref} are defined as

$$\Delta_\theta(x, y^+, y^-) = \log \pi_\theta(y^+ | x) - \log \pi_\theta(y^- | x), \quad (2)$$

$$\Delta_{\text{ref}}(x, y^+, y^-) = \log \pi_{\text{ref}}(y^+ | x) - \log \pi_{\text{ref}}(y^- | x). \quad (3)$$

Text-to-image diffusion models have become the leading framework for high-fidelity image generation from natural language inputs. These models iteratively denoise random noise into coherent images conditioned on text prompts, leveraging powerful encoders such as CLIP [28]. Foundational works like *DALL-E 2* [31], *Imagen* [34], and *Stable Diffusion* [32] have shown remarkable capabilities in producing semantically aligned and visually diverse images. These systems benefit from training on large-scale datasets using latent diffusion architectures that balance quality and efficiency. Extensions such as *InstructPix2Pix* [5] and *SDEdit* [23] introduce mechanisms for editing existing images through text-based instructions. However, these approaches remain general-purpose and do not adapt to individual users’ stylistic preferences, limiting their applicability in personalized editing scenarios.

4 Methodology

We introduce Collaborative Direct Preference Optimization (C-DPO), a novel training framework that extends DPO by incorporating user-specific embeddings and graph-based collaboration for personalized image editing. Section 4.1 details our collaborative loss, which aligns edits with both individual and neighbor preferences, along with our user conditioning and training strategy. Section 4.2 describes personalized image editing via our diffusion-based backend. Fig. 2 gives an overview of our methodology.

4.1 Collaborative Direct Preference Optimization (C-DPO)

Direct Preference Optimization (DPO) has recently emerged as a stable and efficient alternative to RLHF for aligning models with human preferences. However, in its standard form, DPO is fundamentally limited for the task of personalized image editing. First, it lacks cross-user collaboration: all preference pairs are treated as if originating from a single anonymous user, leading the model to converge toward an average policy that ignores the diverse and often overlapping aesthetic preferences of real users. Second, as shown in Eq. 1, DPO assumes unimodal textual conditioning, which restricts its ability to incorporate structured, non-textual information such as user relationships or historical preferences encoded in a knowledge graph. These shortcomings prevent standard DPO from leveraging either personalization or collaboration, both of which are essential for effective user-specific image editing.

4.1.1 Modeling User-Edit Preferences in a Graph

To overcome the limitations of vanilla DPO, we extend the editing policy to explicitly condition on user identity and incorporate collaborative structure through a graph-based model of user preferences. We define the personalized editing policy as:

$$\pi_{\theta}(y \mid u, I_c, t), \quad (4)$$

where $u \in \mathcal{U}$ denotes the user identifier, I_c is the caption describing the base image, and t is the high-level editing instruction. Each user is represented by a learnable embedding $g_{\phi}(u) \in \mathbb{R}^d$, which encodes the user’s stylistic preferences and conditions the model to generate personalized outputs.

To enable collaboration across users with overlapping preferences, we construct a heterogeneous, bipartite, undirected user–preference graph:

$$\mathcal{G} = (\mathcal{U}, \mathcal{P}, \mathcal{E}), \quad (5)$$

where \mathcal{U} is the set of users initialized with a learnable embedding vector, \mathcal{P} denotes preference attributes (e.g., color palettes, textures) initialized with a dense text embedding derived from a pretrained language model, and \mathcal{E} consists of edges between users and their preferred or rejected attributes. This structure captures shared editing behavior and supports collaborative inference through shared interactions.

We encode the graph using a lightweight graph neural network (GNN), which aggregates neighbor information to compute a contextualized embedding h_u for each user [12, 17]. Our GNN consists of a 2-layer GraphSAGE encoder and a decoder composed of 2 linear layers. We pretrain the GNN on an auxiliary supervised edge classification task by predicting user–attribute links as positive or negative, encouraging the encoder to learn semantically meaningful and stylistically distinct user embeddings.

A core benefit of our approach is that it generalizes to new users without requiring retraining, enabling both scalability and practicality. GraphSAGE inductively learns a set of aggregation functions that takes as input a node’s neighborhood. When an unseen user arrives at inference, they are first added as a node with a zero-initialized feature vector in the existing graph and connected via edges to liked and disliked attributes. This new user’s neighborhood is passed through the learned aggregation functions to compute the user’s node embedding. While periodic fine-tuning as the graph expands may further refine embedding quality, it is not necessary for inference. Further GNN details can be found in Appendix B.

To quantify similarity between users, we compute a normalized similarity score between users u and v based on the number of shared one-hop neighbors (i.e., common preference attributes):

$$w_{uv} = \frac{|P(u) \cap P(v)|}{\max_{u', v'} |P(u') \cap P(v')|}, \quad (6)$$

where $P(u) \subseteq \mathcal{P}$ is the set of attributes associated with user u . Here, $P(u)$ denotes the set of preference attributes connected to user u , and the denominator is the maximum number of shared neighbors across all user pairs in the graph. These scores are later used as edge weights in our collaborative preference objective, allowing the model to softly attend to information from stylistically similar users. By conditioning the editing policy on graph-informed embeddings and user identity, our framework supports both fine-grained personalization and generalization through collaborative structure, addressing the core deficiencies of traditional DPO for personalized image editing.

4.1.2 Collaborative DPO Loss

To align the editing policy with both individual user preferences and insights from like-minded users, we introduce a collaborative extension to the DPO objective. The core idea is to preserve strong per-user alignment while softly regularizing the model using preference signals from a user’s graph neighbors.

For a mini-batch of preference tuples, (u, I_c, t, y^+, y^-) we define the Collaborative Direct Preference Optimization (C-DPO) loss as:

$$\mathcal{L}_{\text{C-DPO}} = \underbrace{\mathcal{L}_{\text{DPO}}(u, I_c, t, y^+, y^-)}_{\text{individual}} + \frac{\lambda}{\sum_{v \in \mathcal{N}(u)} w_{uv}} \sum_{v \in \mathcal{N}(u)} w_{uv} \underbrace{\mathcal{L}_{\text{DPO}}(v, I_c, t, y^+, y^-)}_{\text{collaborative}} \quad (7)$$

where $\mathcal{N}(u)$ denotes the K nearest neighbors of u in \mathcal{G} , and λ governs the strength of collaboration. Weighted averaging ensures that the collaborative term does not overpower the individual objective. Each \mathcal{L}_{DPO} is instantiated via 1, but with the policy now conditioned on the *user information*. Because the reference policy π_{ref} remains *user-agnostic*, the collaborative term pulls the user-conditioned policy toward neighbor preferences *only where the data supports it*, avoiding degenerate collapse to the global average. Equation 7 generalizes DPO to a *multi-user* setting. The first term preserves the per-user alignment benefits of vanilla DPO, whereas the second term implements a graph-regularized collaborative filter that shares statistical strength among like-minded users—crucial for data-sparse personalization.

4.1.3 Model Optimization Strategy

We adopt a two-stage training pipeline. First, we perform supervised fine-tuning (SFT) on paired base image and editing instruction data [5] to teach a pretrained language model to generate high-quality editing instructions conditioned on the image caption I_c and text instruction t , but without any user information. Specifically, we train a LoRA adapter [15] on this task, enabling efficient adaptation of the base LLM for instruction generation. The resulting adapter is frozen and serves as the reference model π_{ref} during subsequent preference-based training.

Afterwards, we fine-tune a separate copy of the same LoRA adapter using our collaborative DPO objective (Equation 7) over our user preference dataset. This adapter serves as the policy model π_θ .

To personalize the model, we inject user-specific information as soft prompt tokens. For each user u , we compute a graph-based embedding h_u using our GNN encoder. These embeddings are passed through a two-layer MLP and converted into a fixed set of soft tokens, which are prepended to the textual instruction t . This approach enables user conditioning without modifying the base language model architecture. Both the GNN and MLP components are trainable during C-DPO along with the LoRA, though we apply a lower learning rate to the GNN to preserve the structure learned during pretraining. During C-DPO training, the model learns not only from each user’s preferences but also from the preferences of graph-connected neighbors, promoting better generalization across similar users while preserving individual alignment.

4.2 Image Editing with Personalized Editing Prompt

Once a personalized editing prompt is generated by our C-DPO model, we translate it into a concrete pixel-level transformation using a modular diffusion-based editing backend. We utilize FLUX [18], a large-scale rectified flow transformer developed for high-fidelity text-to-image generation, augmented with ControlNet [42] to ensure faithful, spatially constrained edits.

For a target user u , input image caption I_c , and high-level textual cue t , our tuned C-DPO model outputs a *personalized prompt*:

$$p_u = f_\theta(u, I_c, t),$$

where f_θ denotes the tuned C-DPO policy model. The output p_u is a natural-language instruction tailored to user u ’s preferences, and is directly consumable by any T2I-style sampler. For example: “change the sofa’s color to hot pink.”. This setup enables faithful and personalized image edits that reflect the user’s intent with high semantic precision.

5 Experiments

We evaluate our personalized-editing framework with extensive qualitative and quantitative experiments. Qualitative experiments showcase how edits are personalized based on different users, while quantitative experiments measure how well the edited images reflect each user’s unique style and preferences across thousands of (image, user) pairs.

Experimental Setup We evaluate our method’s ability to generate personalized edits for a given base image. To model user-specific edit instructions, we fine-tune QWEN2.5 [27] using LoRA [15]. For image editing, we employ FLUX.1-DEV², enhanced with ControlNet³ using a conditioning scale of 0.4 and Canny as the condition. We set the collaborative term strength, λ , to 0.15. While our full pipeline can be trained on a single NVIDIA L40 (48GB) GPU, we use multiple GPUs to accelerate training. During the initial supervised fine-tuning phase, we train for one epoch on the InstructPix2Pix training set [5] using 2 GPUs, completing in just under 3 hours. For collaborative tuning via C-DPO, we train for 3000 steps on our user editing preference dataset using 3 GPUs, with total training time of approximately 1 hour. We divide our dataset into training and test splits, with approximately 2,900 users in the training set and 100 users reserved for testing. Once trained, our system performs a personalized image edit for a given user in around one minute. Because our framework is editing-model agnostic, this process can be accelerated to about one second when paired with a faster backend such as TurboEdit [10]. Full training details and parameter configurations are provided in Appendix G.

Dataset To the best of our knowledge, no existing dataset captures the user-specific preferences necessary for personalized image editing. Therefore, we present a structured benchmark of 3,000 synthetic user profiles for studying personalized image editing in T2I diffusion models, spanning 144K samples on individual like/dislike preferences. To closely mimic the diversity of behavior, preferences and aesthetic tastes in the real world, we synthetically generate our data with distinct user demographic configurations that cover axes such as age, geography, and socioeconomic status. These configurations serve as a guide to the generation of detailed editing preferences, including themes, tones, overlays, likes, dislikes, and persona-aligned prompt examples. To simulate realistic editing interactions, each user is assigned four randomly sampled image captions from 80 COCO [21] categories. For each base image, we generate two pairs of preferred and rejected editing instructions across six edit types based on individual user preferences, yielding 48 annotated editing instructions per user. This rich data set serves as the foundation for training and evaluating our collaborative personalization framework. Our data set enables a rigorous evaluation of user alignment and preference modeling in diffusion-based editors. Our dataset is publicly available at <http://personalized-editing.github.io>.

5.1 Qualitative Experiments

We first showcase how identical objects are edited in distinct ways according to individual user preferences. Fig. 1, Fig. 3 (a) and Fig. 4 show several objects edited for different users with diverse profiles. Our system aligns each edit based on the likes and dislikes that the user has previously expressed. For instance, for a user (*labeled as ‘Imaginative Child’*) who favors unicorns, rainbows, and vibrant, playful palettes (see Fig. 3 (a) and Fig. 4), our system infuses such preferences into a wide range of objects, from helmets and guitars to watchtowers. Our framework is also flexible to allow users to provide additional guidance while performing personalized edits. Fig. 3 (b) illustrates three distinct additional editing instructions such as *ice, lava, and monster* themes. As shown, our framework effectively combines user-specific preferences with the supplied guidance, producing coherent edits that align with both the explicit instruction and the underlying aesthetic of each user.

5.2 Quantitative Experiments

Since no prior work directly tackles personalized image editing, we evaluate our proposed framework along two complementary dimensions: a) Prompt-level alignment, measuring how well the editing instruction aligns with each user’s liked and disliked concepts, b) Image-level alignment, assessing how closely the resulting edited image reflects the user’s preferences based on user’s liked and disliked concepts.

Prompt-level alignment We compare our method on a diverse set of strong baselines that vary in personalization capability, modeling assumptions, and training objectives. This comparison allows us to isolate the contributions of user conditioning and collaborative learning.

First, we provide two generic vision-language baselines: Qwen [27], a vanilla large-scale language model that takes only the image description as input and generates a generic edit prompt without awareness of user identity; and LLaVA [22], another open-source vision-language model that simi-

²<https://huggingface.co/black-forest-labs/FLUX.1-dev>

³<https://huggingface.co/InstantX/FLUX.1-dev-Controlnet-Union>

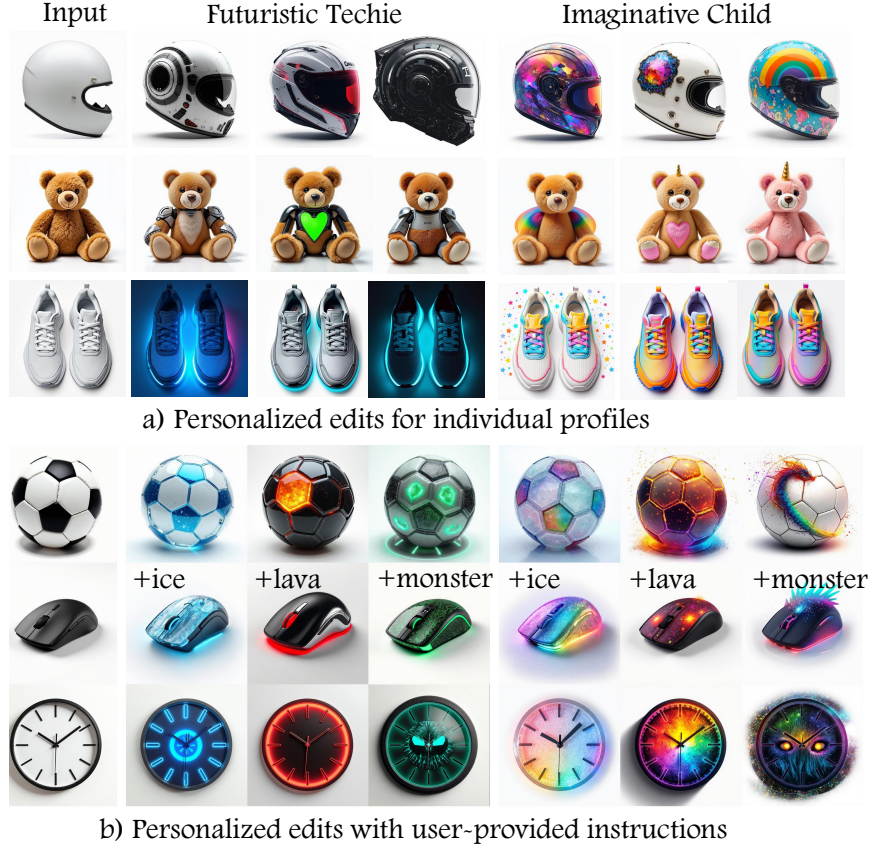


Figure 3: **(a) Qualitative Results for Individual Users on Different Objects.** Our framework is able to incorporate personalized elements into the image editing process, such as adding neon or futuristic elements for *Futuristic Techie* profile. **(b) Qualitative Results for User-Provided Personalized Edits.** Our framework allows users to provide additional guidance while performing personalized edits.

larly lacks any personalization signal and serves as a second reference point for non-user-specific performance. Both models act as task-agnostic baselines with no preference alignment.

Next, we consider Qwen (SFT), a fine-tuned variant trained on 150,000 generic edit-instruction pairs from the InstructPix2Pix dataset [5]. While this model is editing-aware, it lacks user conditioning, allowing us to assess the effect of task-specific fine-tuning alone. We also compare with Vanilla DPO [30], trained on pairwise preference data sampled from our dataset but without access to user identifiers or relational information. This model tests whether learning from global preference signals alone provides any personalization benefit. To evaluate the impact of incorporating explicit user identity, we also compare with a user-aware variant, DPO-User, that uses a learnable embedding for each user ID but does not leverage any collaborative structure. This setup mimics the PPD objective [7] and represents a personalization mechanism based on private embeddings alone. Finally, our proposed method - Collaborative-DPO extends DPO-User by introducing graph-based collaboration, allowing user representations to evolve through interactions with stylistically similar neighbors. This model constitutes the full version of our framework.

To compare these methods fairly, we standardize the prompting format across all models. Each method receives an input of the form: “Given this user’s profile <profile>, suggest an edit for the following image: <image description>.” We construct three evaluation conditions reflecting different profile completeness: (i) Like+Dislike, where both liked and disliked examples are available; (ii) Likes-only; and (iii) Dislikes-only. For each condition, we measure CLIP-text similarity between the generated instruction and the user’s ground-truth liked instructions. As shown in Table 2, our C-DPO

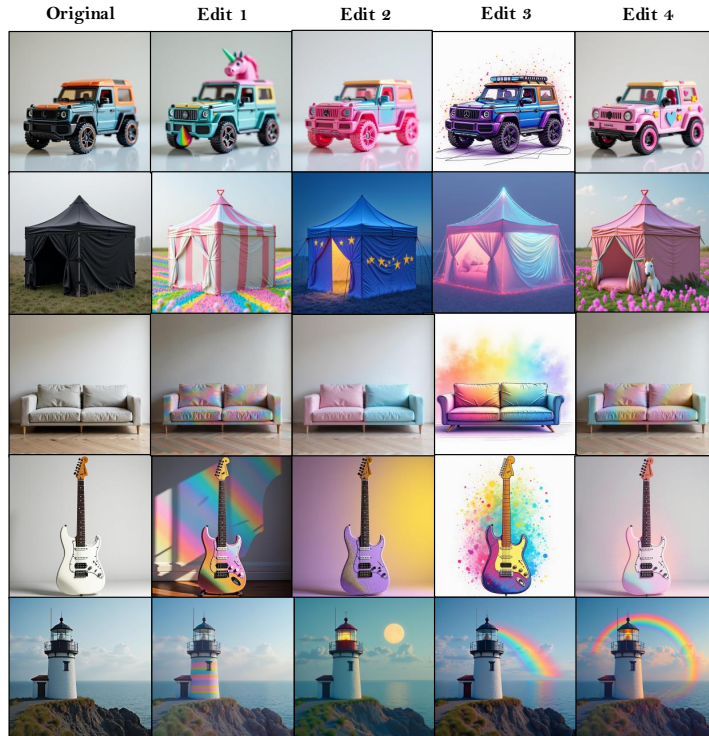


Figure 4: **Qualitative Results for a Single User on Diverse Objects.** For a user who loves unicorns, rainbows, and vibrant, playful palettes, our system infuses that whimsical aesthetic into a wide range of objects - from cars and guitars to watchtowers.

model achieves the highest score across all configurations, indicating its ability to recover user intent from partial or complete preference signals.

Image-level alignment Next, we report DINO and CLIP scores computed between the original input image and its personalized edited version (see Table 3). The results indicate that our method better preserves the original identity compared to other approaches. We also compute HPS and CLIP-T scores between the personalized edited images and the textual description of the original image, further demonstrating that our method achieves higher alignment with the original content than competing methods. Please refer to Appendix A for additional quantitative experiments and comparisons with state-of-the-art editing methods.

Ablations Please refer to Appendix C for ablations on collaborative scale parameter λ , neighborhood size K , choice of editing method, and prompt-engineering baseline.

Personalized Image Generation Our method can also be repurposed for personalized image *generation* as opposed to *editing*. Please visit A.2 for more details.

5.3 User Study

We further validate our results through a user study to assess perceptual personalization quality. We ran a crowd-sourced evaluation on Prolific.com with 50 participants. We randomly sampled 10 synthetic user profiles from our test split to cover a range of stylistic tastes. For each profile we first showed a short text description of what aspects this user likes and dislikes based on their previous edit history. Participants then judged three target images that had been edited by three competing methods (Vanilla DPO, DPO-User and Ours). Each edited image was displayed alongside its text prompt; the order of methods and the order of questions were fully randomized per task to prevent position bias.

Model	Q1	Q2
DPO-Vanilla	0.210	0.210
DPO-User	0.174	0.325
Ours	0.616	0.465

Table 2: CLIP scores between model outputs and user preferences under different prompting conditions where we provided both Like and Dislike preferences of the user (Like+Dislike), only like preferences (Likes) or only dislike preferences (Dislikes).

Model	Like+Dislike	Likes	Dislikes
Qwen-VL	0.274 ± 0.055	0.276 ± 0.052	0.229 ± 0.051
LLaVA	0.244 ± 0.065	0.254 ± 0.061	0.227 ± 0.051
SFT-Vanilla	0.276 ± 0.049	0.276 ± 0.042	0.185 ± 0.065
SFT-Pix2Pix	0.227 ± 0.078	0.238 ± 0.065	0.196 ± 0.065
DPO-Vanilla	0.272 ± 0.059	0.274 ± 0.057	0.172 ± 0.057
DPO-User	0.307 ± 0.071	0.286 ± 0.071	0.269 ± 0.076
Ours	0.354 ± 0.068	0.306 ± 0.069	0.294 ± 0.071

Table 3: DINO and CLIP scores computed between the original input image and its personalized edited version (DINO-Ref, CLIP-Ref). HPS and CLIP-T scores are computed between the personalized edited images and the textual description of the original image.

Model	DINO-Ref	CLIP-Ref	HPS	CLIP-T
SFT	0.690 ± 0.263	0.600 ± 0.222	0.238 ± 0.044	0.331 ± 0.075
DPO-Vanilla	0.736 ± 0.248	0.615 ± 0.202	0.242 ± 0.049	0.346 ± 0.067
DPO-User	0.762 ± 0.240	0.649 ± 0.224	0.243 ± 0.048	0.354 ± 0.066
Ours	0.782 ± 0.182	0.652 ± 0.182	0.249 ± 0.034	0.358 ± 0.043

Adapted from the user study of [38], workers were asked the following questions for every image: Q1: General Preference – “Which edited image would this user prefer, given the prompt?” Q2: Visual Appeal – “Ignoring the prompt, which image is more visually appealing?”. Table 1 summarizes responses from the 50 participants across the two evaluation questions. In every category our method secures the highest win rate by a large margin. These results indicate that, human judges consistently favor the edits produced by our method. We further conduct a user study with real-user preference data, replacing the synthetic preferences used in earlier experiments. In this study, participants provided their own liked and disliked concepts, and we evaluated how well the edited images aligned with these individual preferences. Please refer to Appendix E for detailed methodology and results.

6 Discussion

Broader Impact and Limitations By aligning text-to-image diffusion models to each individual’s aesthetic, our framework can lower the barrier to high-quality visual content creation, reducing repetitive prompt-engineering cycles and enabling non-experts, including artists with motor impairments or limited technical skills to achieve their desired edits more efficiently. On the negative side, since our system tailors outputs to inferred tastes, it also risks reinforcing aesthetic “filter bubbles,” narrowing users’ exposure to diverse visual styles. In terms of technical limitations, when a new user lacks both personal edits *and* close neighbors in the graph, our model defaults to a generic editing. Furthermore, because our framework depends on off-the-shelf models such as Flux and ControlNet, any biases embedded in those backbones may propagate to the resulting edits. As for data, we choose to generate our data synthetically because of the scalability and budget constraints of crowd-sourcing, however, obtaining real-world user preference data presents an exciting future research direction by gathering more authentic and nuanced information.

Conclusion We presented the first framework that unifies *personal* and *collaborative* preference learning for editing with text-to-image diffusion models. Our novel objective function C-DPO (i) models each user’s preference through learned graph embeddings, (ii) propagates information across a similarity graph of like-minded users, and (iii) couples the resulting personalized prompt with a T2I diffusion model to deliver high-fidelity, user-aligned edits. Our study demonstrates that collaborative signals can be harnessed *without* sacrificing individual customization, paving the way for practical editing assistants that adapt fluidly to diverse aesthetics. Future research will focus on extending the framework beyond still images such as *video domain*, where temporal coherence and multi-object consistency introduce new challenges, which remains an exciting direction for follow-up work.

References

- [1] Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F.L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al.: Gpt-4 technical report. arXiv preprint arXiv:2303.08774 (2023)
- [2] Alayrac, J.B., Donahue, J., Luc, P., Miech, A., Barr, I., Hasson, Y., Lenc, K., Mensch, A., Millican, K., Reynolds, M., Ring, R., Rutherford, E., Cabi, S., Han, T., Gong, Z., Samangoei, S., Monteiro, M., Menick, J., Borgeaud, S., Brock, A., Nematzadeh, A., Sharifzadeh, S., Binkowski, M., Barreira, R., Vinyals, O., Zisserman, A., Simonyan, K.: Flamingo: a visual language model for few-shot learning (2022), <https://arxiv.org/abs/2204.14198>
- [3] Avrahami, O., Fried, O., Lischinski, D.: Blended latent diffusion. ACM transactions on graphics (TOG) **42**(4), 1–11 (2023)
- [4] Brack, M., Friedrich, F., Kornmeier, K., Tsaban, L., Schramowski, P., Kersting, K., Passos, A.: Ledits++: Limitless image editing using text-to-image models. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 8861–8870 (2024)
- [5] Brooks, T., Holynski, A., Efros, A.A.: Instructpix2pix: Learning to follow image editing instructions. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 18392–18402 (2023)
- [6] Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J.D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al.: Language models are few-shot learners. Advances in neural information processing systems **33**, 1877–1901 (2020)
- [7] Dang, M., Singh, A., Zhou, L., Ermon, S., Song, J.: Personalized preference fine-tuning of diffusion models. arXiv preprint arXiv:2501.06655 (2025)
- [8] DeepMind, G.: Gemini 1: A family of highly capable multimodal models. Technical Report (2023), <https://deepmind.google/technologies/gemini/>
- [9] Deng, C., Zhu, D., Li, K., Gou, C., Li, F., Wang, Z., Zhong, S., Yu, W., Nie, X., Song, Z., Shi, G., Fan, H.: Emerging properties in unified multimodal pretraining (2025), <https://arxiv.org/abs/2505.14683>
- [10] Deutch, G., Gal, R., Garibi, D., Patashnik, O., Cohen-Or, D.: Turboedit: Text-based image editing using few-step diffusion models. In: SIGGRAPH Asia 2024 Conference Papers. pp. 1–12 (2024)
- [11] Ge, Y., Zhao, S., Zeng, Z., Ge, Y., Li, C., Wang, X., Shan, Y.: Making llama see and draw with seed tokenizer (2023), <https://arxiv.org/abs/2310.01218>
- [12] Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. Advances in neural information processing systems **30** (2017)
- [13] Herlocker, J., Konstan, J.A., Riedl, J.: An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms. Information retrieval **5**(4), 287–310 (2002)
- [14] Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. Advances in neural information processing systems **33**, 6840–6851 (2020)
- [15] Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W.: Lora: Low-rank adaptation of large language models. arXiv preprint arXiv:2106.09685 (2021)
- [16] Junseong Kim, Seolhwa Lee, J.K.S.G.Y.K.M.C.J.y.S.C.C.: Linq-embed-mistral: elevating text retrieval with improved gpt data through task-specific control and quality refinement. Linq AI Research Blog (2024), <https://getlinq.com/blog/linq-embed-mistral/>
- [17] Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
- [18] Labs, B.F.: Flux. <https://github.com/black-forest-labs/flux> (2024)

- [19] Li, X., Liu, Z., Guo, S., Liu, Z., Peng, H., Yu, P.S., Achan, K.: Pre-training recommender systems via reinforced attentive multi-relational graph neural network. In: 2021 IEEE international conference on big data (big data). pp. 457–468. IEEE (2021)
- [20] Li, X., Zhou, R., Lipton, Z.C., Leqi, L.: Personalized language modeling from personalized human feedback (2024), <https://arxiv.org/abs/2402.05133>
- [21] Lin, T.Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., Perona, P., Ramanan, D., Zitnick, C.L., Dollár, P.: Microsoft coco: Common objects in context (2015), <https://arxiv.org/abs/1405.0312>
- [22] Liu, H., Li, C., Wu, Q., Lee, Y.J.: Visual instruction tuning (2023), <https://arxiv.org/abs/2304.08485>
- [23] Meng, C., He, Y., Song, Y., Song, J., Wu, J., Zhu, J.Y., Ermon, S.: Sdedit: Guided image synthesis and editing with stochastic differential equations. arXiv preprint arXiv:2108.01073 (2021)
- [24] Nabati, O., Tennenholtz, G., Hsu, C., Ryu, M., Ramachandran, D., Chow, Y., Li, X., Boutilier, C.: Personalized and sequential text-to-image generation. arXiv preprint arXiv:2412.10419 (2024)
- [25] Oquab, M., Darcet, T., Moutakanni, T., Vo, H., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., Massa, F., El-Nouby, A., Assran, M., Ballas, N., Galuba, W., Howes, R., Huang, P.Y., Li, S.W., Misra, I., Rabbat, M., Sharma, V., Synnaeve, G., Xu, H., Jegou, H., Mairal, J., Labatut, P., Joulin, A., Bojanowski, P.: Dinov2: Learning robust visual features without supervision (2024), <https://arxiv.org/abs/2304.07193>
- [26] Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al.: Training language models to follow instructions with human feedback. *Advances in neural information processing systems* **35**, 27730–27744 (2022)
- [27] Qwen, :, Yang, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Li, C., Liu, D., Huang, F., Wei, H., Lin, H., Yang, J., Tu, J., Zhang, J., Yang, J., Yang, J., Zhou, J., Lin, J., Dang, K., Lu, K., Bao, K., Yang, K., Yu, L., Li, M., Xue, M., Zhang, P., Zhu, Q., Men, R., Lin, R., Li, T., Tang, T., Xia, T., Ren, X., Ren, X., Fan, Y., Su, Y., Zhang, Y., Wan, Y., Liu, Y., Cui, Z., Zhang, Z., Qiu, Z.: Qwen2.5 technical report (2025), <https://arxiv.org/abs/2412.15115>
- [28] Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., Sutskever, I.: Learning transferable visual models from natural language supervision (2021), <https://arxiv.org/abs/2103.00020>
- [29] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I.: Language models are unsupervised multitask learners. *OpenAI Blog* **1**(8), 9 (2019)
- [30] Rafailov, R., Sharma, A., Mitchell, E., Manning, C.D., Ermon, S., Finn, C.: Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems* **36**, 53728–53741 (2023)
- [31] Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., Chen, M.: Hierarchical text-conditional image generation with clip latents (2022), <https://arxiv.org/abs/2204.06125>
- [32] Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. pp. 10684–10695 (2022)
- [33] Rout, L., Chen, Y., Ruiz, N., Caramanis, C., Shakkottai, S., Chu, W.S.: Semantic image inversion and editing using rectified stochastic differential equations. arXiv preprint arXiv:2410.10792 (2024)
- [34] Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E.L., Ghasemipour, K., Gontijo Lopes, R., Karagol Ayan, B., Salimans, T., et al.: Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems* **35**, 36479–36494 (2022)

- [35] Salehi, S., Shafiei, M., Yeo, T., Bachmann, R., Zamir, A.: Viper: Visual personalization of generative models via individual preference learning. In: European Conference on Computer Vision. pp. 391–406. Springer (2024)
- [36] Song, J., Meng, C., Ermon, S.: Denoising diffusion implicit models (2022), <https://arxiv.org/abs/2010.02502>
- [37] Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, S., Batra, S., Bhargava, P., Bhosale, S., et al.: Llama 2: Open foundation and fine-tuned chat models. arXiv preprint arXiv:2307.09288 (2023)
- [38] Wallace, B., Dang, M., Rafailov, R., Zhou, L., Lou, A., Purushwalkam, S., Ermon, S., Xiong, C., Joty, S., Naik, N.: Diffusion model alignment using direct preference optimization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8228–8238 (2024)
- [39] Wu, X., Hao, Y., Sun, K., Chen, Y., Zhu, F., Zhao, R., Li, H.: Human preference score v2: A solid benchmark for evaluating human preferences of text-to-image synthesis (2023), <https://arxiv.org/abs/2306.09341>
- [40] Xu, Y., Wang, W., Zhang, Y., Tang, B., Yan, P., Feng, F., He, X.: Personalized image generation with large multimodal models. In: Proceedings of the ACM on Web Conference 2025. pp. 264–274 (2025)
- [41] Zhang, H., Duan, Z., Wang, X., Zhao, Y., Lu, W., Di, Z., Xu, Y., Chen, Y., Zhang, Y.: Nexus-gen: Unified image understanding, generation, and editing via prefilled autoregression in shared embedding space (2025), <https://arxiv.org/abs/2504.21356>
- [42] Zhang, L., Rao, A., Agrawala, M.: Adding conditional control to text-to-image diffusion models. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 3836–3847 (2023)
- [43] Zhou, X., Lin, D., Liu, Y., Miao, C.: Layer-refined graph convolutional networks for recommendation. In: 2023 IEEE 39th international conference on data engineering (ICDE). pp. 1247–1259. IEEE (2023)

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: We clearly state our claims in the introduction and abstract and support our claims through qualitative and quantitative experimentation.

Guidelines:

- The answer NA 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 NA 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: We include a separate "Limitations" section.

Guidelines:

- The answer NA 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: [NA]

Justification: We do not include theoretical results.

Guidelines:

- The answer NA 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: We thoroughly discuss our method and provide detailed hyperparameter and training details with the supplementary material. The code, dataset, and model checkpoint are also provided with the supplementary material.

Guidelines:

- The answer NA 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: [Yes]

Justification: We provide data, code and usage instructions in the supplementary material.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.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://nips.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, etc.) necessary to understand the results?

Answer: [\[Yes\]](#)

Justification: We highlight important experimental settings with the main paper (e.g. model choice, hyperparameters for our new loss, run time). We include full training details and hyperparameters in the supplementary material.

Guidelines:

- The answer NA 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: [\[Yes\]](#)

Justification: We provide standard deviation with all quantitative experiments.

Guidelines:

- The answer NA 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: We provide gpu information, resource usage for both training and inference, and train and run time in experimental setup.

Guidelines:

- The answer NA 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: We reviewed and meet all aspects in the NeurIPS Code of Ethics.

Guidelines:

- The answer NA 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: We discuss broader impacts of our work.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA 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., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: Our current framework does not have high risk and data is synthetically generated.

Guidelines:

- The answer NA 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: We properly cite all code, models and datasets that we leverage in our work.

Guidelines:

- The answer NA 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: [Yes]

Justification: Our provided code and dataset are well documented to support reproducibility clarity.

Guidelines:

- The answer NA 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: We show screenshots of questions asked in our user study in the supplementary and mention the paid platform where it was conducted.

Guidelines:

- The answer NA 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: [No]

Justification: Our conducted user study does not pose such risks, so we did not describe them in the paper.

Guidelines:

- The answer NA 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: We use LLMs for caption generation and dataset curation in our framework and this is clearly described in the paper.

Guidelines:

- The answer NA 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 (<https://neurips.cc/Conferences/2025/LLM>) for what should or should not be described.

Table of Contents

A	Additional Quantitative Experiments	21
A.1	Experiments on Personalized Image Editing	21
A.2	Experiments on Personalized Image Generation	21
A.3	Comparison with State-of-the-Art Instruction-Based Editing Models	21
B	Detailed GNN Information	22
B.1	GNN Construction and Training	22
B.2	Generalizing to Unseen Users	22
C	Additional Ablations	23
C.1	Ablation on Collaborative Scale Parameter	23
C.2	Ablation on Neighborhood Size	23
C.3	Ablation on Editing Method	23
C.4	Ablation on Prompt-Engineering Baseline	23
D	Additional Qualitative Experiments	24
D.1	Additional Qualitative Examples	24
D.2	Visuals on Personalized Image Generation	24
D.3	Comparison with State-of-the-Art Instruction-Based Editing Models	24
E	Additional User Studies	25
F	Evaluation Prompt	26
G	Experimental Details	26

Table 4: We report pairwise similarity scores between the user’s liked images and the generated personalized images (image–image similarity using DINO-Pref and CLIP-I-Pref), as well as between the personalized edited images and the user’s liked prompts (image–text similarity using CLIP-T-Pref).

Model	DINO-Pref	CLIP-I-Pref	CLIP-T-Pref	CLIP-T-Neighbor
SFT	0.054 ± 0.102	0.315 ± 0.098	0.334 ± 0.049	0.286 ± 0.025
DPO-Vanilla	0.056 ± 0.106	0.335 ± 0.097	0.359 ± 0.035	0.304 ± 0.026
DPO-User	0.056 ± 0.105	0.325 ± 0.096	0.359 ± 0.034	0.297 ± 0.026
Ours	0.059 ± 0.106	0.338 ± 0.089	0.364 ± 0.038	0.312 ± 0.020

Table 5: Quantitative comparison for personalized image generation with VIPER.

Model	CLIP-I	DINO	HPS
ViPer	0.286 ± 0.015	0.035 ± 0.008	0.205 ± 0.050
Ours	0.326 ± 0.020	0.037 ± 0.016	0.253 ± 0.046

A Additional Quantitative Experiments

We conduct additional quantitative experiments to evaluate our personalized image editing framework. Specifically, we compare user-preferred images with the personalized outputs generated by our method using DINO [25], CLIP-I, CLIP-T [28], and HPS [39] metrics (see Sec. A.1).

Furthermore, we demonstrate the effectiveness of our approach for personalized image generation by benchmarking against state-of-the-art methods in this domain (see Sec. A.2).

A.1 Experiments on Personalized Image Editing

Table 2 (main paper) presents a quantitative comparison between our method and other state-of-the-art approaches using editing prompts generated by each method, as these baselines only support prompt-level generation rather than direct image outputs. To further evaluate image-level quality, we conduct additional experiments comparing SFT, Vanilla DPO, and User-based DPO against our method. For each method, we generated editing prompts and used Flux ControlNet to produce corresponding images. Results of the quantitative comparisons are shown in Table 4 and Table 3.

First, in Table 4, we report pairwise similarity scores between the user’s liked images and the generated personalized images (image–image similarity using DINO-Pref and CLIP-I-Pref), as well as between the personalized edited images and the user’s liked prompts (image–text similarity using CLIP-T-Pref). We also include a CLIP-T-Neighbor measure, which evaluates how well the personalized images align with attributes favored by similar users. For each user, we identify their ten most similar users based on the user-preference graph, aggregate the those users liked attributes, and compute the CLIP similarity between these aggregated preferences and the images generated for the target user. As can be seen from the results, our method achieves better results in generating personalized editing images.

A.2 Experiments on Personalized Image Generation

To further validate the effectiveness of our method beyond personalized image editing, we compare it with the state-of-the-art personalized image generation method VIPER [35]. As shown in Table 9 and Fig. 9, our method achieves higher scores when comparing generated images to the user’s liked images. Additionally, it delivers superior visual quality and relevance, better aligning with user preferences (see Fig. 9).

A.3 Comparison with State-of-the-Art Instruction-Based Editing Models

To better contextualize our approach within the broader landscape of instruction-based and unified image-editing frameworks, we conducted additional quantitative comparisons against recent state-of-

Table 6: Quantitative comparison with recent state-of-the-art instruction-based editing frameworks. Our C-DPO method achieves superior personalization and overall alignment

Method	CLIP-I-Pref	DINO-Pref	HPS
BAGEL	0.3397 ± 0.0144	0.0363 ± 0.0066	0.2475 ± 0.0388
SEED-Llama	0.3109 ± 0.0159	0.0349 ± 0.0121	0.2500 ± 0.0287
Nexus-Gen	0.3327 ± 0.0138	0.0373 ± 0.0113	0.2598 ± 0.0307
Ours	0.3515 ± 0.0190	0.0365 ± 0.0164	0.2607 ± 0.0328

the-art baselines: BAGEL [9], SEED-LLAMA [11], and Nexus-Gen [41]. For fairness, we provided each model with the same textual user-preference prompts used in our setup, thereby allowing them to leverage equivalent conditioning information without requiring architectural modification.

As summarized in the Table 9, our approach clearly outperformed these methods across standard image editing evaluation metrics, highlighting the distinct advantage of our collaborative personalization framework.

B Detailed GNN Information

Here, we go into more detail regarding graph architecture, GNN construction and training, and generalization to unseen users at inference.

B.1 GNN Construction and Training

We employ a lightweight Graph Neural Network (GNN) to compute contextualized user embeddings by aggregating neighborhood information in a heterogeneous bipartite graph of users and attributes. Each attribute node represents an editing instruction topic (e.g., rustic themes, neon colors, minimalistic textures) and is initialized with a dense text embedding derived from a pretrained language model. Each user node is initialized with a learnable embedding vector. The resulting graph encodes user-attribute relationships via edges that indicate whether a user likes or dislikes a given attribute.

Our GNN architecture consists of a 2-layer GraphSAGE convolutional encoder using mean aggregation [12], followed by an edge decoder comprising two linear layers. Details on specific layer dimensions can be found in Table 12. The model is pretrained on an auxiliary edge-prediction task designed to classify whether a user-attribute link is positive or negative. This task encourages the GNN to learn semantically meaningful user representations grounded in the structure of the user-attribute graph. We train the model on 60% of the edges, reserving 20% each for validation and testing, following standard protocols outlined in foundational works [12, 17].

During pretraining, both liked and disliked user-attribute pairs are leveraged in a supervised edge classification task (e.g., edges of the form (user1, likes, “neon green”) or (user2, dislikes, “rainbows’’)). Once pretraining is complete, the classification head is discarded and only the encoder’s user embeddings are retained. These embeddings are subsequently used in the full Collaborative Direct Preference Optimization (C-DPO) stage, where user embeddings are combined with target image captions (e.g., “White ceramic bowl”) to construct user-specific positive and negative instruction pairs (e.g., “Add a holographic pattern to the bowl” as positive, “Overlay a floral pattern on the bowl” as negative).

B.2 Generalizing to Unseen Users

A central advantage of our approach is that it supports generalization to new users without retraining. Our approach follows an inductive embedding framework (GraphSAGE) rather than a transductive one. Unlike transductive methods, which are limited to fixed graphs, GraphSAGE learns a set of aggregation functions that generalize to unseen nodes by aggregating features from their neighborhoods. This allows our model to efficiently infer embeddings for new users.

At inference time, when a new users arrives, (1) we create a new user node with a zero-initialized feature vector of fixed width, (2) we connect this node to relevant attribute nodes representing the user’s liked and disliked attributes. (3) We apply the pretrained GNN encoder’s learned aggregation

Table 7: Ablation on C-DPO Neighborhood Size (K)

K	CLIP Liked Similarity
2	0.353 ± 0.065
3	0.354 ± 0.068
5	0.358 ± 0.065
12	0.344 ± 0.069

functions over this new node’s neighborhood to compute the user’s embedding inductively from the connected attributes and their neighborhood structure.

This process requires no retraining, making it scalable and practical for continual deployment. Although periodic fine-tuning on the expanded graph may further refine embedding quality, it remains optional and is not necessary for inference.

C Additional Ablations

C.1 Ablation on Collaborative Scale Parameter

We conduct an ablation study on the collaborative scaling factor λ , evaluating values of 0.01, 0.15, and 0.50 (see Fig. 7). A very low value ($\lambda = 0.01$) limits the influence of neighbor preferences, effectively reducing the method to a user-based DPO, as neighbors contribute insufficient information to guide personalization. In contrast, a high value ($\lambda = 0.50$) overly emphasizes neighbor preferences, which can override the user’s own preferences and degrade performance. Our results indicate that $\lambda = 0.15$ provides the best balance, allowing meaningful influence from similar users without overwhelming the target user’s preferences.

C.2 Ablation on Neighborhood Size

We further investigate the effect of the collaborative neighborhood size K , which determines how many of the most similar users contribute weighted preference signals in the C-DPO loss 7. As shown in Table 7, smaller to moderate values yield stronger alignment between generated edits and user preferences, as measured by CLIP Like Similarity. Increasing K beyond a certain threshold leads to diminishing returns or even degraded performance, likely due to noise from less relevant neighbors. Although the synthetic nature of our dataset may influence this, the phenomenon is not exclusive to synthetic data and generally occurs in collaborative and graph-based methods, even with real-world datasets [13, 43, 19]. When too many neighbors are aggregated, weaker or less relevant signals tend to dilute the informative collaborative signals from closely related neighbors—a well-documented issue across various real-world collaborative filtering and recommendation tasks.

C.3 Ablation on Editing Method

In our experiments, we used Flux ControlNet as the base editing method. However, our method is applicable to Stable Diffusion-based editing methods as well. To highlight the impact of the underlying editing method, we compared personalized editing results across several Stable Diffusion-based approaches: SDEdit [23], Ledits++ [4], InstructPix2Pix [5], TurboEdit [10] as well as Flux-based methods including RF Inversion [33] and Flux ControlNet⁴. As shown in Fig. 8, Flux ControlNet consistently delivers superior fidelity and visual quality compared to the other Flux-based methods. Similarly, our personalized editing with Ledits++ performs better than other SD based methods.

C.4 Ablation on Prompt-Engineering Baseline

To further justify the added complexity of our GNN-based collaborative personalization framework, we compare C-DPO against a strong prompt-engineering baseline that directly injects textual user descriptions, including aggregated neighbor information, into the model prompt, without using

⁴<https://huggingface.co/InstantX/FLUX.1-dev-Controlnet-Union>

Table 8: Comparison between our C-DPO framework and a prompt-engineering baseline that encodes user and neighbor information textually. Our method achieves markedly higher user-alignment scores, validating the effectiveness of GNN-based personalization.

Method	Like+Dislike	Likes	Dislikes
Prompt Engineering	0.270 ± 0.051	0.270 ± 0.045	0.200 ± 0.057
Ours	0.354 ± 0.068	0.306 ± 0.069	0.294 ± 0.071

Table 9: Quantitative comparison with recent state-of-the-art instruction-based editing frameworks. Our C-DPO method achieves superior personalization and overall alignment

Method	CLIP-I-Pref	DINO-Pref	HPS
BAGEL	0.3397 ± 0.0144	0.0363 ± 0.0066	0.2475 ± 0.0388
SEED-Llama	0.3109 ± 0.0159	0.0349 ± 0.0121	0.2500 ± 0.0287
Nexus-Gen	0.3327 ± 0.0138	0.0373 ± 0.0113	0.2598 ± 0.0307
Ours	0.3515 ± 0.0190	0.0365 ± 0.0164	0.2607 ± 0.0328

GNN-derived embeddings. This baseline represents a simplified but potentially competitive approach where user and neighborhood preferences are expressed through structured textual conditioning rather than learned representations.

As shown in Table 8, our proposed method substantially outperforms this baseline across all metrics. These results indicate that our proposed framework provides a meaningful benefit over simpler, prompt-based personalization methods, effectively justifying the additional complexity introduced by the GNN.

D Additional Qualitative Experiments

D.1 Additional Qualitative Examples

We first showcase how identical objects are edited in distinct ways according to individual user preferences. Fig. 5 shows several objects edited for different users with diverse profiles. Our system adapts each image to the likes and dislikes that the user has previously expressed (see ‘User Profile’ which describes user’s preferences). For instance, the first row depicts a user who prefers playful elements, vivid color palettes and childish elements such as unicorns and rainbows. When this user is presented with a bike or a ceramic bowl, our model automatically adds rainbow-hued paint, pastel gradients, and subtle unicorn-themed playful touches that align with a child-friendly aesthetic, while preserving the underlying object structure in a disentangled manner. On the other hand, when our system is presented with a user who prefers earthy tones and nature, our framework adds natural elements such as leaves, wooden materials and plants.

D.2 Visuals on Personalized Image Generation

We further demonstrate the applicability of our method for personalized image generation by comparing it with the state-of-the-art approach, VIPER [35]. Fig. 9 presents results for two distinct user profiles: the Imaginative Child, who prefers rainbow colors and glitter, and the Glamour Stylist, who favors shiny and metallic elements. As shown, our method captures a broader range of user-specific preferences while maintaining focus on coherent objects. In contrast, VIPER often emphasizes textures or patterns without anchoring them to meaningful objects.

D.3 Comparison with State-of-the-Art Instruction-Based Editing Models

To better contextualize our approach within the broader landscape of instruction-based and unified image-editing frameworks, we conducted additional quantitative comparisons against recent state-of-the-art baselines: BAGEL [9], SEED-LLAMA [11], and Nexus-Gen [41]. For fairness, we provided each model with the same textual user-preference prompts used in our setup, thereby allowing them to leverage equivalent conditioning information without requiring architectural modification.





















User Profile	Original	Edit 1	Edit 2	Edit 3	Edit 4
<p>The user is drawn to bright, whimsical colors, fantastical elements, and playful overlays like sparkles, rainbows, or unicorn motifs. They favor soft, glowing visuals and imaginative edits, while rejecting muted tones, minimalist styles, and anything overly realistic or dull.</p>					
<p>This user prefers images with earthy tones, natural textures, and a calm, organic feel. They favor leaf and floral overlays, soft lighting, and wood or stone materials, while avoiding neon colors, synthetic textures, and digital or cartoon effects. Their edits evoke a grounded, nature-inspired aesthetic.</p>					
<p>The user prefers images that convey luxury and elegance, with vibrant tones, glamorous lighting, and editorial balance. Their aesthetic favors high-contrast detail, bokeh, and refined textures like polished metal or marble, while rejecting grunge, cartoon, and floral overlays in favor of a sleek, fashion-forward look.</p>					
<p>The user prefers images with cosmic textures, warped forms, and vivid, altered color schemes. They favor dreamlike overlays, double exposures, and abstract compositions that evoke surrealism, while rejecting basic filters, clean commercial edits, and literal or minimalist styles.</p>					

Figure 5: **Qualitative Results for Different Users on the Same Objects.** Our framework tailors image edits to each user’s preferences. In the first row, for example, a user who loves unicorns, rainbows, and playful color schemes sees their inputs transformed to match that personalized aesthetic preferences.

As summarized in the Table 9, our approach clearly outperformed these methods across standard image editing evaluation metrics, highlighting the distinct advantage of our collaborative personalization framework.

E Additional User Studies

We conducted a two-stage user survey with 10 participants using real-human data. Specifically, we first asked participants to indicate their likes and dislikes attributes from a predefined list of 45 concepts as suggested. This list covers a wide range of colors (e.g. standard colors, neon colors), styles (e.g. watercolor, cyberpunk), textures (e.g. glossy, wood) and other concepts (such as rainbow, butterfly).

Based on these preferences, we generated personalized images using our proposed method. Participants were then sent a follow-up survey featuring personalized edits of five objects: an apple, a mug, a bag, a teddy bear, and a bowl. In this survey, participants viewed both the original and personalized versions of each object and answered the following questions:

Q1: On a scale of 1 to 5 (1=Does not matching my personal taste at all, 5=This is matching my personal taste a lot) can you rate the following image?

Q2: To what extent does the edit preserve the original image while reflecting personalized preferences? (1=Not at all, 5=Very Much)

Table 10: Additional User Study results.

Object Name	Q1: Personalization	Q2: Disentanglement
Apple	4.375 ± 0.74	4.75 ± 0.46
Mug	4.625 ± 1.06	4.50 ± 0.75
Teddy bear	4.75 ± 0.46	4.625 ± 0.51
Bag	4.50 ± 1.06	3.85 ± 0.83
Bowl	4.75 ± 0.46	5.00 ± 0.00
Total	4.60 ± 0.15	4.55 ± 0.38



Figure 6: Example edit for a member of our real user study. *This user preferred pink, purple, and neon colors, while disliking white, gray, and beige. Their favored patterns included rainbow, fluffy, and glossy styles, while they disliked floral and earthy designs.* This image was rated as Personalization = 5/5, Disentanglement = 4/5, showing that the user found the edit matching their taste.

Our results (Table 10) demonstrate that real users found the images personalized by our framework highly personalized and disentangled.

On the same real-user preference data, we conducted an additional user study with personalized edits for each image to compare baselines. Specifically, users are presented with an object alongside three personalized edits (from Vanilla DPO, User-DPO, and our C-DPO method) where images are shown in an anonymous fashion. Users are asked to pick the edit they most prefer out of 3 options.

Table 11: Additional User Study comparison of preference ratios across models.

Object	C-DPO (Ours)	User-DPO	User-Vanilla
A Bag	0.45	0.22	0.33
A Bowl	1.00	0.00	0.00
A Mug	0.45	0.33	0.22
A Teddy Bear	0.78	0.22	0.00
An Apple	1.00	0.00	0.00

F Evaluation Prompt

To ensure a fair comparison across baselines, we adopt a unified structured prompt for all evaluations (see Fig. 10).

G Experimental Details

Various details of our experimental setup are given in Table 12.

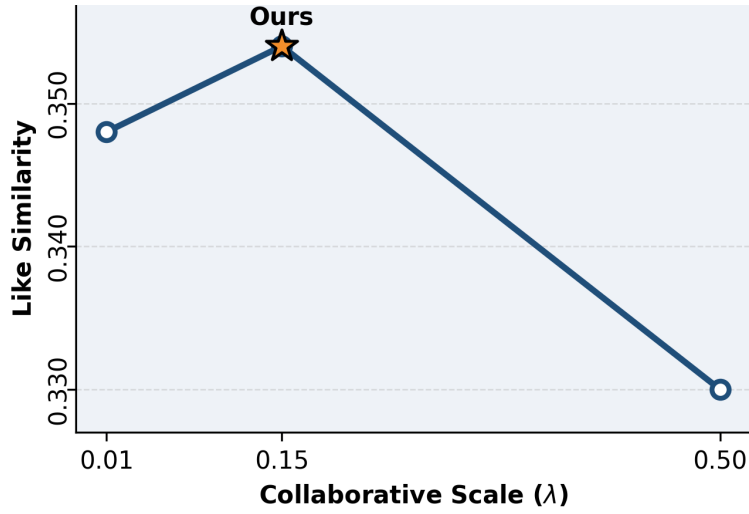


Figure 7: Ablation on Collaborative Scale (λ)



Figure 8: Ablation on Stable Diffusion and Flux based editing methods.

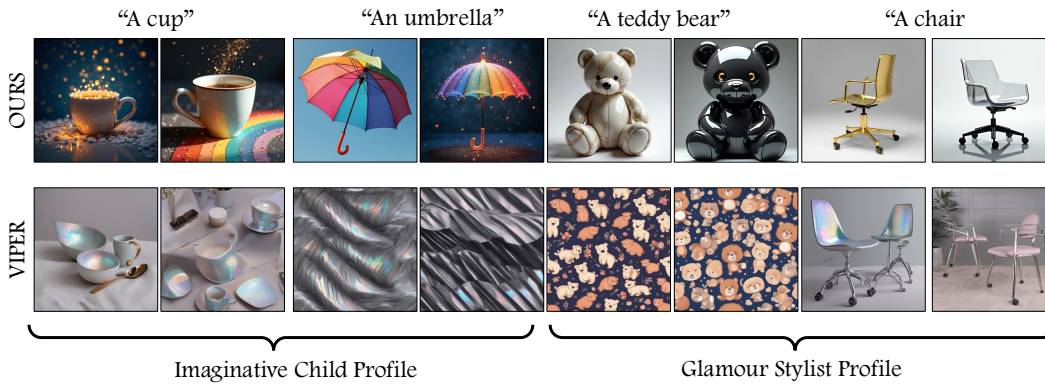


Figure 9: Personalized image generation for two profiles: Imaginative Child Profile who likes rainbow colors and glitter, and Glamour Stylist Profile who likes shiny and metallic elements.

System Prompt

You are an expert at providing user personalized image-editing instructions.

User Prompt

Likes: [e.g. colorful and playful styles, unicorn motifs, rainbows]
 Dislikes: [e.g. minimalistic styles, muted tones, overly realistic imagery]

Original Image Caption: [e.g. "A white ceramic bowl"]

Suggest a single, cohesive image-editing instruction.

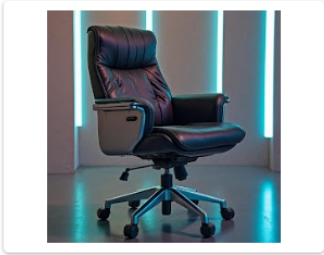
Figure 10: The input prompt template used during our evaluations.

Which edited image would this user prefer, given their preferences? *

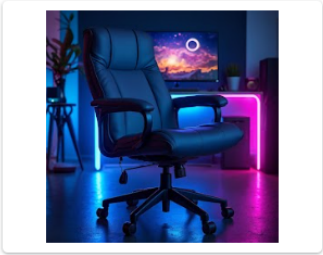
User Preferences:

👍 **Likes:**
 Futuristic | Sci-fi | Neon colors | Glass & metal textures | Holographic & circuit patterns | LED/glow effects | Cyberpunk/digital styles

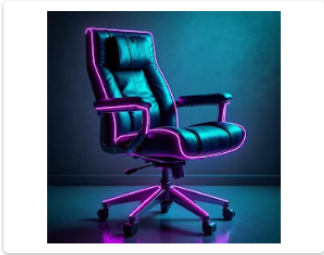
👎 **Dislikes:**
 Vintage | Sepia/pastels | Wood/fabric textures | Floral/polka patterns | Traditional art styles



Option 2



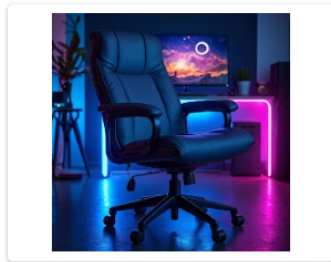
Option 3



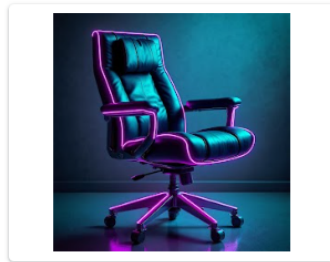
Option 1

Figure 11: User Study Q1

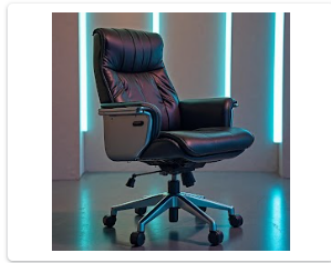
Which image is more visually appealing? *



Option 3



Option 1



Option 2

Figure 12: User Study Q2

LoRA Details	
Base LLM	Qwen/Qwen2.5-7B-Instruct [27]
LoRA Rank	16
LoRA α	32
Dropout	0.05
Optimizer	AdamW
LR Scheduler	Cosine
Data Type	bfloat16
Target Modules	All Linear
User Based GNN	
Embedding Size	4096
Keyword Encoder	Linq-AI-Research/Linq-Embed-Mistral [16]
Pretrain Epochs	150
Pretrain Learning Rate	3e-5
Pretrain Optimizer	Adam
GraphSAGE Hidden Channels	1024
Linear Hidden Channels	1024
Initial Supervised Finetuning	
Data	Instruct Pix2Pix [5]
GPUs	2 NVIDIA L40 (48GB) GPUs
Train Time	~3 Hours
Epochs	1
Learning Rate	2e-6
Batch Size	16
Collaborative DPO	
Data	Our Synthetic Data
GPUs	3 NVIDIA L40 (48GB) GPUs
Start Checkpoint	SFT Tuned LoRA on Instruct Pix2Pix
Train Time	~1 Hour
Number of Similar Users	3
Collaborative Term Scale (λ)	0.15
Number of Soft Prompt Tokens	8
Train Steps	3000
Learning Rate	2e-7
GNN Parameters Learning Rate	2e-8
Batch Size	12
Flux+ControlNet	
Flux Model	black-forest-labs/FLUX.1-dev [18]
ControlNet	InstantX/FLUX.1-dev-Controlnet-Union
Control Mode	Canny
Number of Inference Steps	60
Guidance Scale	35
ControlNet Conditioning Scale	0.3-0.4

Table 12: Full Experimental Details for Replicating Our Method