Hollowed Net for On-Device Personalization of Text-to-Image Diffusion Models

Wonguk Cho[*,1,2] **Seokeon Choi**[1] **Debasmit Das**[1] **Matthias Reisser**[1]
Taesup Kim[2] **Sungrack Yun**[1] **Fatih Porikli**[1]

[1]Qualcomm AI Research[†] [2]Seoul National University

[1]{wongcho, seokchoi, debadas, mreisser, sungrack, fporikli}@qti.qualcomm.com
[2]{wongukcho, taesup.kim}@snu.ac.kr

Abstract

Recent advancements in text-to-image diffusion models have enabled the personalization of these models to generate custom images from textual prompts. This paper presents an efficient LoRA-based personalization approach for on-device subject-driven generation, where pre-trained diffusion models are fine-tuned with user-specific data on resource-constrained devices. Our method, termed Hollowed Net, enhances memory efficiency during fine-tuning by modifying the architecture of a diffusion U-Net to temporarily remove a fraction of its deep layers, creating a *hollowed* structure. This approach directly addresses on-device memory constraints and substantially reduces GPU memory requirements for training, in contrast to previous methods that primarily focus on minimizing training steps and reducing the number of parameters to update. Additionally, the personalized Hollowed Net can be transferred back into the original U-Net, enabling inference without additional memory overhead. Quantitative and qualitative analyses demonstrate that our approach not only reduces training memory to levels as low as those required for inference but also maintains or improves personalization performance compared to existing methods.

1 Introduction

Recent research on text-to-image (T2I) diffusion models [1, 2], which generate high-resolution images from text prompts, has increasingly focused on personalizing and customizing these generative models effectively [3, 4, 5, 6, 7]. A primary approach, termed subject-driven generation [5], involves fine-tuning pre-trained diffusion models with a few user-specific images to generate varied representations of a subject using simple text prompts. This allows users to create personalized images of specific subjects, such as family, friends, pets, or personal items, with preferred appearances, backgrounds, and styles. Such capabilities enable creative applications including art renditions, property modifications, and accessorization.

From a practical standpoint, implementing subject-driven generation on-device offers significant benefits in efficiency and privacy. By operating independently of congested cloud servers or networks, users can generate personalized images anywhere at no additional cost and do not need to compromise their privacy as all data and personal information remain on the device.

[*]Work done during an internship at Qualcomm AI Research.
[†]Qualcomm AI Research is an initiative of Qualcomm Technologies, Inc.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

Despite extensive research aimed at efficiently personalizing diffusion models, limited attention has been paid to memory I/O, a critical bottleneck in on-device learning. Recent studies have mainly explored two strategies: (1) decreasing the number of training steps and (2) reducing the number of updating parameters. The first methods [8, 9, 10, 11, 12] utilize additional large pre-trained models to generate a set of personalized Low-Rank Adaptation (LoRA) parameters [13], text embeddings, or image prompts from a user-specific image. This strategy provides a better initial setup for personalizing the diffusion models, effectively reducing required training steps. Some models [10, 11, 12] even support zero-shot personalization, although they underline that further fine-tuning can enhance personalization quality and address failure cases. Nonetheless, these methods are not viable for environments with severely limited computational resources, as they necessitate additional inference using large pre-trained models (e.g., 2.7B parameters for BLIP-2 in BLIP-Diffusion [12] and 2.5B for apprentice models in SuTI [11]), which are substantially larger than standard diffusion models (e.g., 1B for Stable Diffusion v2 [2]), making their application challenging in on-device settings.

The second approach [7, 14], often involving LoRA, aims to reduce the number of updating parameters by limiting updates to specific layers or decomposing weight matrices. However, even with fewer parameters to update, these parameters reside within large pre-trained models, and thus the backward pass through the large models is required to compute gradients. Given limited computational resources, where even simple inference tasks with diffusion models can strain GPU memory, performing backpropagation while keeping the entire diffusion model in GPU memory remains a significant limitation.

A promising approach to address these challenges is side-tuning [15, 16, 17, 18], which fine-tunes a smaller auxiliary network rather than directly updating the parameters of a large pre-trained network. This method significantly reduces the heavy memory costs associated with computing backpropagation on the larger network. Particularly for Natural Language Processing (NLP) tasks, Ladder Side Tuning (LST) [18] has proven effective, reducing the memory costs required for fine-tuning large language models (LLMs) by 69 percent. However, applying LST directly to diffusion U-Nets presents significant challenges. Unlike transformer layers in LLMs, which maintain consistent input and output dimensions, diffusion U-Nets have varying spatial dimensions and channels, as well as skip-connections across different blocks. Additionally, the requirements for structural pruning and weight initialization to build side-tuning networks further complicate the rapid adaptability of LST to personalization tasks across different subjects and domains.

To this end, we introduce a novel personalization technique called *Hollowed Net*, which is illustrated in Fig. 1. Based on our observation that deep layers in the middle of diffusion U-Nets play significantly less important roles than the rest of the layers, we propose to fine-tune LoRA parameters for the personalization using Hollowed Net, a layer-pruned U-Net featuring a central hollow, which is constructed by temporarily removing the middle deep layers from the pre-trained diffusion U-Net. By utilizing the symmetrical "U-shape" architecture of the diffusion U-Net, we avoid complicated processes of applying structural pruning and weight initialization to build a side network, and neither additional models nor extensive pre-training with large datasets are required.

By fine-tuning LoRA parameters using Hollowed Net, we can significantly reduce the memory needed for storing model weights in GPU. Once the LoRA parameters are fine-tuned with Hollowed Net, they can be seamlessly transferred back to the original Diffusion U-Net for inference, without requiring any additional memory beyond the small set of transferred parameters. Our experiments demonstrate that Hollowed Net enables achieving performance that is comparable to or better than the direct fine-tuning with LoRA, while using 26 percent less GPU memory, which is only 11 percent increased GPU memory relative to an inference.

To the best of our knowledge, Hollowed Net is the first technique that addresses subject-driven generation in terms of memory efficiency. Our method shows how T2I diffusion models can be fine-tuned under extremely limited computational resources with as low GPU memory as required for inference. Furthermore, it is important to note that our method does not preclude the use of previously described strategies for efficient personalization. Both enhanced parameter-efficient strategies and improved initializations with additional pre-trained models can be integrated with our approach to further increase efficiency according to given resource constraints.

Our contributions can be summarized as follows:

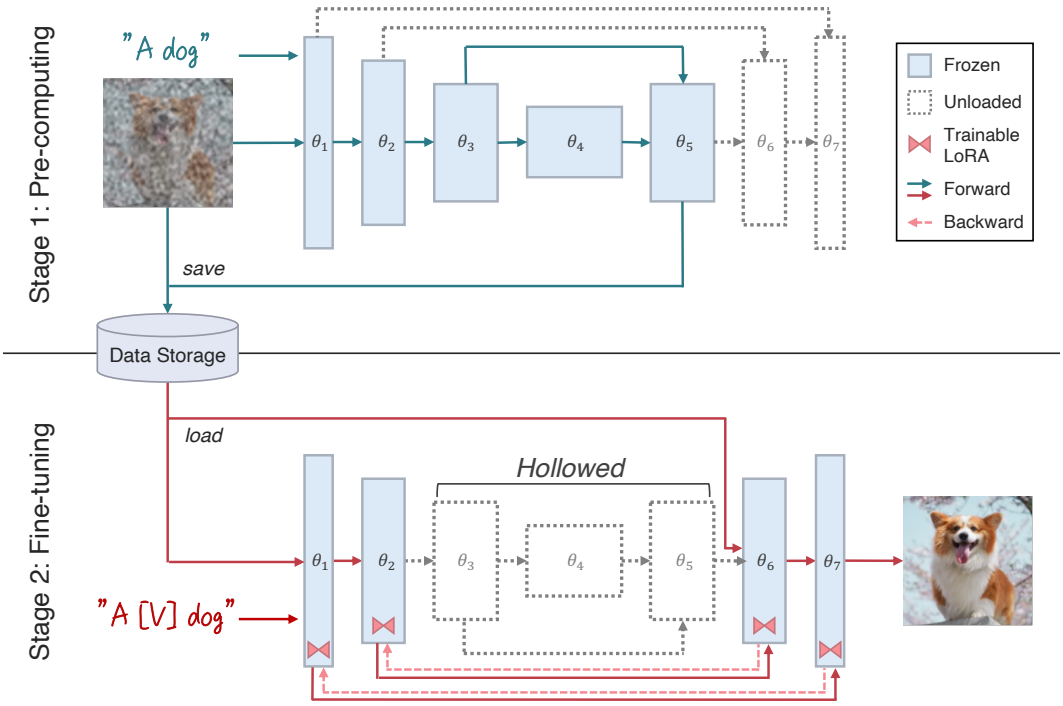


Figure 1: The LoRA personalization with Hollowed Net for resource-constrained environments. The input image is from the DreamBooth dataset [5].

- We introduce *Hollowed Net*, a novel personalization technique for T2I diffusion models under limited computational resources. Our method significantly reduces the memory demands on GPU to levels as low as those required for inference, while maintaining a high-fidelity personalization capacity. This demonstrates its potential as a feasible on-device learning solution for resource-constrained devices.
- Our method provides a scalable and controllable solution for on-device learning. As this method does not require any additional models or pre-training with large datasets, it is easily scalable to other architectures such as SDXL and Transformers. Moreover, we can simply adjust the fraction of hollowed layers to control the trade-offs between performance and memory requirements, depending on the target application and resources.
- Unlike previous side-tuning methods, Hollowed Net does not need to be retained for inference. The LoRA parameters fine-tuned with Hollowed Net can be seamlessly transferred back to its original network, enabling inference with no additional memory cost.

2 Related Works

2.1 Efficient Personalization of T2I Diffusion Models

Recent research on the personalization of T2I diffusion models has introduced various methods to fine-tune the models for generating diverse images of user-specific subjects from a few given images. Two foundational works in this area are Textual Inversion and DreamBooth [3, 5]. Textual Inversion [3] aims to learn new text embeddings to represent a given subject, while DreamBooth [5] proposes fine-tuning an entire diffusion model to align the subject with a unique token.

Building on these foundational works, recent research has focused on enhancing the efficiency of this personalization process, primarily through two approaches. The first approach involves decreasing the number of training steps, mostly by utilizing an additional large pre-trained model. A popular method is to use a pre-trained image/multi-modal encoder to generate personalized text embeddings

or image prompts from a user-specific image [9, 12, 10]. Other recent works [8, 11] propose utilizing a set of pre-optimized LoRA parameters or millions of fine-tuned expert models to pre-initialize for efficient fine-tuning or enable zero-shot generation with in-context learning. While these models demonstrate significant reductions in the number of training steps, the requirement for additional large pre-trained models limits their application to on-device settings. Moreover, models with zero-shot personalization capacities [11, 12, 10] cannot be a one-size-fits-all solution for addressing different types of user-subject prompts. These models often struggle with flexibility in editing subjects or maintaining subject fidelity, and in these cases, additional fine-tuning with specific subjects is needed to further enhance their personalization capacity [11, 12].

On the other hand, another stream of work adapts parameter-efficient fine-tuning (PEFT) approaches. These methods demonstrate significant reductions in the number of training parameters by limiting updates to a small subset of model weights in cross-attention layers [7] or further reducing the updating parameters by applying singular vector decomposition to weight matrices [14]. However, these methods are still limited in environments with extremely low computational resources, as they require backpropagation over large diffusion models and do not reduce memory usage from the model weights. Therefore, it is crucial to explore new approaches for personalizing T2I diffusion models in resource-limited settings, as we propose with our novel method, Hollowed Net. Notably, our method can be integrated with previously discussed techniques to further improve efficiency based on specific resource constraints.

2.2 Fine-Tuning with Side Networks

The idea of of side-tuning has been introduced by Zhang et al. [15], proposing the training of a lightweight "side" network instead of directly fine-tuning a pre-trained network for adaptation. In terms of efficiency, Cai et al. [16] has demonstrated an additional lightweight residual module can reduce memory overhead associated with the activations of the original network. Similarly, AuxAdapt [17] has shown that a small auxiliary network can be fine-tuned to adjust the main network's decisions, enabling efficient test-time adaptation for video semantic segmentation tasks.

In the context of generative models, LST [18] has demonstrated the effectiveness of side networks for different NLP tasks with LLMs by introducing a small side network that takes intermediate activations of the main network as input via shortcut connections. However, directly applying LST to diffusion U-Nets poses challenges due to varying spatial dimensions, channel sizes, and skip-connections across blocks, unlike the consistent dimensions in transformer layers of LLMs. Furthermore, the structural pruning and specific weight initialization required to construct side-tuning networks complicate LST's adaptability for personalized tasks across a range of subjects and domains.

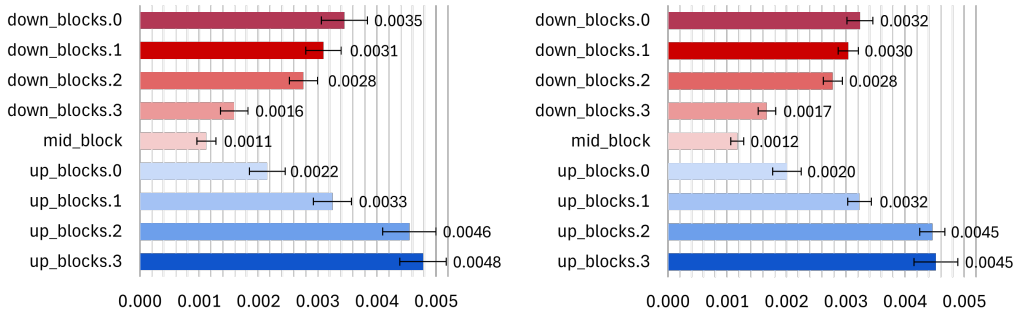
2.3 Layer Pruning of Large Generative Models

Several concurrent works demonstrate that layer-pruning methods can be applied to generative models, particularly for NLP tasks. Gromov et al. [19] suggest that for fine-tuning LLM models, up to 40% of deep layers can be removed, while still achieving comparable results. The authors propose that the optimal block of layers to prune can be selected based on similarity across layers. Similarly, Kim et al. [20] also propose a depth-pruning approach by evaluating block-level importance.

These approaches differ from ours due to the distinct characteristics of LLMs versus diffusion U-Nets. The aforementioned approaches involve the complete removal of deep layers for both fine-tuning and inference, considering that those layers store less critical knowledge. However, our study finds that the deep layers of diffusion U-Nets may be less involved with personalization but still contain crucial high-level image features for generating high-fidelity images. Thus, their removal can lead to severe performance degradation, even with additional pre-training [21], as shown in Appendix A. This highlights the importance of our two-stage fine-tuning strategy, which excludes layers during fine-tuning to reduce memory overhead while preserving the knowledge from these excluded layers throughout both training and inference stages.

3 Preliminaries

In this section, we describe some preliminaries on T2I diffusion models. First, we discuss the basics of Stable Diffusion (SD) model and how they can be used for fine-tuning. The SD model is a large



(a) DreamBooth (b) CustomConcept101

Figure 2: Analysis of the LoRA weight change before and after personalization, per block of U-Net.

foundational T2I model, pre-trained on large amount of text-image pairs (P, x), where we have image x and associated text prompt P. The SD contains the following components: (a) Autoencoder consisting of the encoder-decoder pair $(\mathcal{E}, \mathcal{D})$, (b) Text Encoder as CLIP $E_T(\cdot)$, and (c) Conditional Diffusion Model as U-Net $\epsilon_\theta(\cdot)$. The encoder $\mathcal{E}(\cdot)$ processes an image x into a latent space $z = \mathcal{E}(x)$, and the decoder is used to reconstruct the input image from latent z such that $x \approx \mathcal{D}(z)$. The diffusion process of SD is conducted in the latent space. For a randomly sampled noise $\epsilon \sim \mathcal{N}(0, I)$ at time step t, the standard scheduler produces a noisy latent code $z_t = \alpha_t z + \sigma_t \epsilon$, where α_t and σ_t are coefficients controlling the noise schedule. The conditional diffusion model ϵ_θ is trained using the following de-noising objective:

$$\texttt{min} \quad \mathbb{E}_{P,z,\epsilon,t}[||\epsilon - \epsilon_\theta(z_t, t, E_T(P))||_2^2]. \tag{1}$$

After the training is carried out, the conditioned model $\epsilon_\theta(\cdot)$ is used to predict the noise by using the conditional embedding $E_T(P)$ and time step t as input. To personalize diffusion models for subject-driven generation introduced by [5], the same loss is used except that the data is sampled from user-specific subjects such as dog, person, backpack, and etc. For the prompt, a special identifier $S*$ is used and described as "a $S*$ person", "a $S*$ backpack", etc. For regularization, [5] introduces an additional class-specific prior preservation loss term, written as

$$\texttt{min} \quad \mathbb{E}_{z,\epsilon,t}[||\epsilon_{pr} - \epsilon_\theta(z'_t, t, E_T(P_{pr}))||_2^2] \tag{2}$$

where ϵ_{pr} is the ground truth noise for the data generated using the frozen pre-trained diffusion model with prompts P_{pr} described more generic as "a person", "a backpack", and etc.

The diffusion U-Net can be fully fine-tuned, but it is also possible to fine-tune only a subset of parameters with LoRA [13] for better efficiency. In LoRA, network weight residuals ΔW are fine-tuned instead of the full weights W. For the fine-tuning of ΔW, it is further decomposed into low-rank matrices A and B such that $\Delta W = AB$. Since A and B are low-rank matrices, the total number of parameters to optimize in ΔW is significantly smaller than in W.

4 Methodology

In this section, we describe the details of our novel memory-efficient personalization technique, Hollowed Net, and its fine-tuning strategy. We begin by identifying less significant layers for personalization from diffusion U-Nets. Based on these observations, we explain how to construct Hollowed Net from a pre-trained U-Net. Next, we present our fine-tuning and inference processes for memory-efficient personalization of T2I diffusion models.

4.1 Analysis of the LoRA Weight Changes per Block of U-Net

To achieve the goal of reducing the required memory for fine-tuning a diffusion model, we first identify less significant layers in the diffusion U-Net for personalization. Similar to Li et al. [22], Kumari et al. [7] and Shah et al. [6], we analyze the LoRA weight changes ΔW in the fine-tuned model for each block:

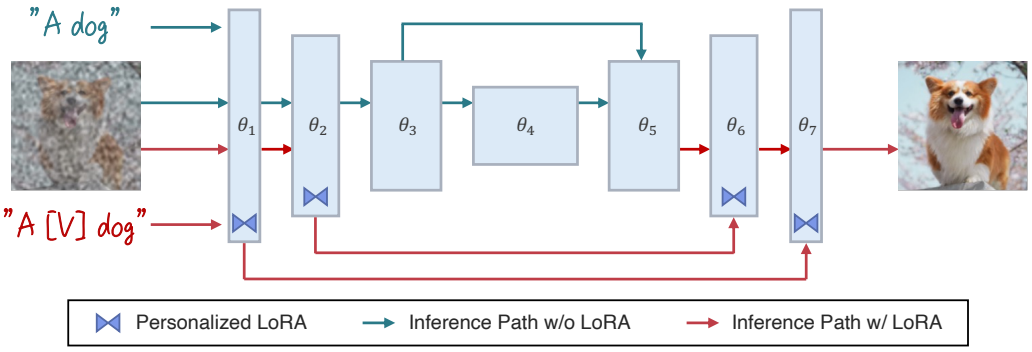


Figure 3: The inference process with personalized LoRA parameters transferred from Hollowed Net to the original U-Net. The input image is from the DreamBooth dataset [5].

$$\Delta W = \frac{1}{n}\sum_{i=1}^{n}|w_i - w_i'|, \tag{3}$$

where w and w' respectively represent the weights before and after personalization, and n is the total number of weights in a specific block. This represents the average weight change per element i. Figure 2 shows the analysis of the weight changes ΔW before and after personalization for each block of U-Net: (a) for all subjects from the DreamBooth dataset and (b) for all subjects from the CustomConcept101 dataset by fine-tuning Stable Diffusion v2.1 diffusion model [2] for 1000 steps with a learning rate of 1e-4. The x-axis shows the changes in LoRA weights before and after personalization, while the y-axis of each plot represents the specific U-Net blocks. For each dataset, we average the weight changes across subjects and provide error bars to indicate the statistical variance within each dataset.

From the figures, we observe that the average weight changes tend to be close to zero around the central blocks and become increasing for the layers farther from the mid_block. This demonstrates that the blocks around the center are less involved in the personalization compared to those at the beginning and end of the U-Net (e.g., down_blocks.0, down_blocks.1, up_blocks.2, and up_blocks.3). We leverage this characteristic for designing Hollowed Net.

4.2 Hollowed Net

Based on the aforementioned observations, we propose fine-tuning a layer-pruned U-Net, which we refer to as Hollowed Net, instead of directly fine-tuning the entire diffusion model. The core concept of Hollowed Net involves removing deep layers that are not vital for personalization from a pre-trained diffusion U-Net. This strategy decreases the need to store the entire model in GPU memory, thereby reducing the memory cost associated with the model's weights.

However, unlike transformer layers in large language models, where input and output maintain the same data structure, the alterations in spatial and channel dimensions in U-Net architectures complicate the removal of its deep layers in the middle. To address this, we utilize the symmetrical "U-shape" architecture of the diffusion U-Net, where each down-block layer's output is concatenated with a corresponding up-block layer's input via a skip-connection. This design permits us to select any up-block layer skip-connected to a down-block layer and hollow out the middle layers between the pair, ensuring that the processed information from the remaining down-blocks can still be transferred to the remaining up-blocks without the need for additional projection layers to adjust for dimensional differences. The missing input for the upper layer, due to the removal of the middle layers, is replaced with the pre-computed output from the full diffusion U-Net, which is illustrated in the next section.

4.3 LoRA Personalization with Hollowed Net

To optimize GPU memory utilization, we propose a two-stage fine-tuning strategy: (1) pre-computing intermediate activations of the original diffusion U-Net and (2) fine-tuning the Hollowed Net using the pre-computed activations, as shown in the upper and bottom half of Fig. 1, respectively. Initially, we

Table 1: The quantitative comparisons of fine-tuning methods with three evaluation metrics. The number of parameters are the ones held in GPU memory during fine-tuning stage. The results are obtained by averaging over four runs with different seeds (standard deviation is added in a small-sized text).

Method	# of Parameters		Training Memory		DreamBooth			CustomConcept101		
	Base	LoRA	Peak	Comp. w/ Inf.	DINO	CLIP-I	CLIP-T	DINO	CLIP-I	CLIP-T
Full FT	866M	-	16.62GB	+376%	0.663 ±0.013	0.802 ±0.007	0.302 ±0.002	0.605 ±0.005	0.773 ±0.006	0.302 ±0.002
LoRA FT (r=128)	866M	27M	5.23GB	+50%	0.658 ±0.001	0.806 ±0.005	0.299 ±0.002	0.603 ±0.008	0.773 ±0.005	0.302 ±0.002
LoRA FT (r=1)	866M	207K	4.84GB	+39%	0.516 ±0.011	0.738 ±0.003	0.314 ±0.001	0.522 ±0.008	0.737 ±0.005	0.305 ±0.001
Hollowed Net (Ours)	**527M**	24M	**3.88GB**	**+11%**	0.660 ±0.011	0.805 ±0.006	0.300 ±0.001	0.603 ±0.007	0.773 ±0.005	0.302 ±0.002

Table 2: Human evaluation results

Method	Subject Fidelity	Text Fidelity
Hollowed Net	31.2%	18.1%
Tie	49.3%	69.4%
LoRA FT	19.5%	12.5%

Table 3: Computational loads (FLOPs)

Method	Pre-computing	Fine-tuning	Inference
Hollowed Net	0.238T	2.004T	0.920T
LoRA FT	-	2.148T	0.716T

conduct a forward pass with a pre-trained diffusion model for the specified number of pre-computing steps. During each step, given input images and sampled noise, we calculate and store intermediate activations in the data storage, which serve as inputs for the upper-block layer of the Hollowed Net. We also store the sampled noises, time steps, and the IDs when there are multiple user images.

Once the data from the pre-trained model is pre-computed, we fine-tune the Hollowed Net by loading data from data storage, thereby avoiding the need to keep the original model in GPU memory. To further improve efficiency, we apply LoRA fine-tuning for the Hollowed Net instead of updating entire parameters. The reduced number of parameters of the Hollowed Net decreases the required GPU memory, satisfying the device's low memory I/O threshold and computational load during backpropagation.

Additionally, our inference process ensures that both the original diffusion model and Hollowed Net are not simultaneously maintained on GPU. Unlike side-tuning networks [15, 17, 18] that differ in architecture and parameters from their original models, Hollowed Net maintains the same architectures and parameters as the original diffusion U-Net, except for the removed middle layers. Thus, the LoRA parameters fine-tuned on Hollowed Net can be seamlessly transferred to the corresponding layers in the original U-Net. As depicted in Fig. 3, there are two inference paths, respectively corresponds to each stage of fine-tuning. The first path (green line) represents the process of computing intermediate activations without using LoRA, aligning with the pre-computing stage. The second path (red line) involves using personalized LoRA parameters, which matches the application of these parameters for generating personalized images during the fine-tuning stage. By sequentially executing these paths, we enable personalized generation using the transferred LoRA parameters without requiring additional memory beyond the small set of LoRA parameters.

5 Experiments

5.1 Experimental Settings

We conduct experiments following the protocol proposed in DreamBooth [5]. We use a total of 131 subjects for experiments, utilizing both the DreamBooth [5] and CustomConcept101 [7] datasets. The DreamBooth dataset includes 30 image sets from 15 different classes, each containing 4-6 images of a given subject. The subjects are divided into living subjects and objects, and 25 different prompts are assigned based on this division. Meanwhile, the CustomConcept101 dataset includes 101 image sets, each containing 3-15 images of a given subject. The subjects consist of 15 different large categories, with 20 unique prompts assigned to each category. For evaluation, four images with different fixed random seeds are generated per subject per prompt for both datasets.

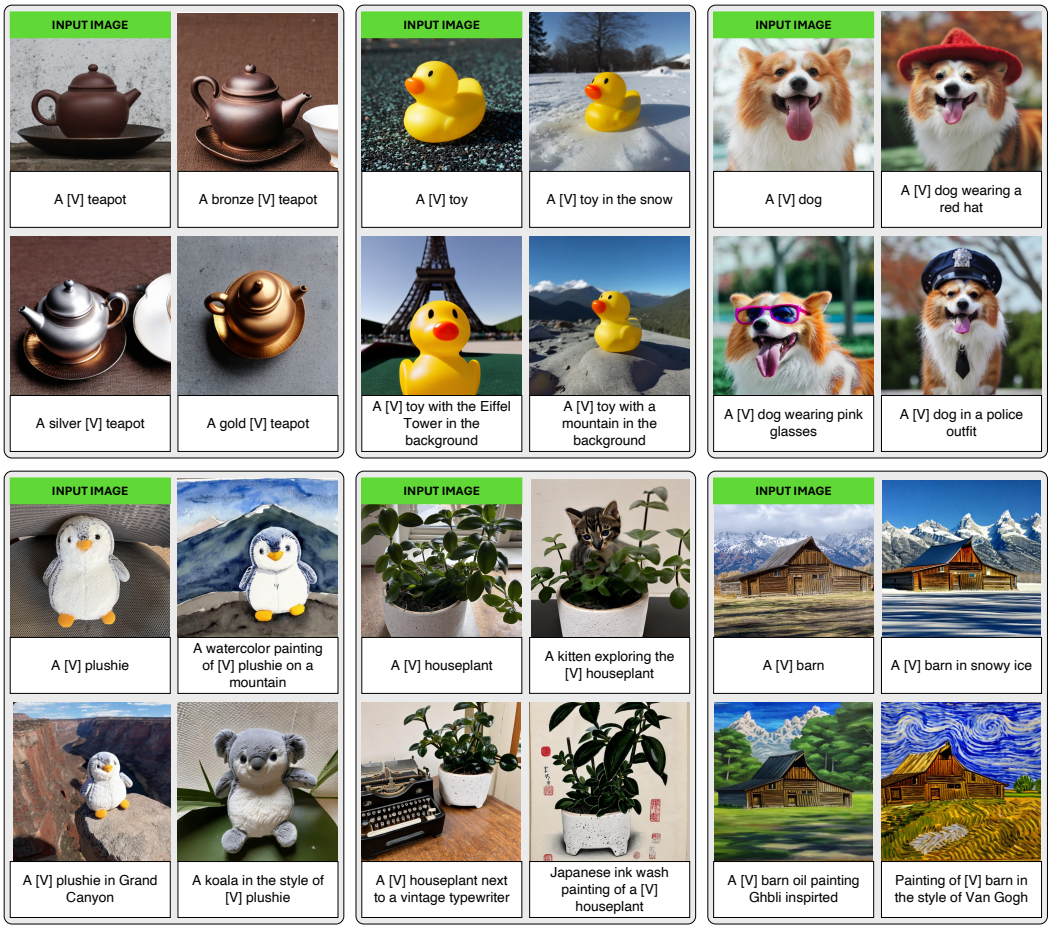


Figure 4: Qualitative generation results of Hollowed Net with different subjects and prompts. The upper half are the examples from the DreamBooth dataset [5], and the lower half are the examples from the CustomConcept101 dataset [7].

We adopt the three evaluation metrics from [5]: DINO and CLIP-I for subject fidelity and CLIP-T for prompt fidelity. DINO and CLIP-I are the average pairwise cosine similarities between feature embeddings of the real and generated images, using DINO ViT-S/16 and CLIP ViT-B/32, respectively. As DINO is more sensitive to differences between subjects of the same class due to its training on instance discrimination, the DINO score is considered the preferred metric for measuring subject fidelity. The CLIP-T score is the average cosine similarity between text prompt embeddings and image CLIP embeddings. We use the Stable Diffusion v2.1 diffusion model [2]. Following DreamBooth [5], we use a prior preservation loss with $\sim$1000 pre-generated class samples. LoRA [13] is applied for the cross and self-attention layers and fine-tuned for $\sim$1000 steps. We use AdamW optimizer with the learning rate of 1e-5 for full-finetuning and 1e-4 for the others. Assuming a resource-constrained environment, we use a batch size of 1 and do not update the pre-trained text encoder, while text embeddings are pre-computed before fine-tuning.

5.2 Results

In this section, we present the results of our proposed Hollowed Net to evaluate its effectiveness in terms of both memory efficiency and personalization performance. We conduct experiments with Hollowed Net, applying a hollowed fraction of 39.2%. Architectural details are provided in Appendix B. Ablation studies on different fractions of hollowed layers can be found in Sec. 5.3. In the main results, the rank of Hollowed Net is fixed to 128. Experimental results on different ranks are presented in Appendix C.

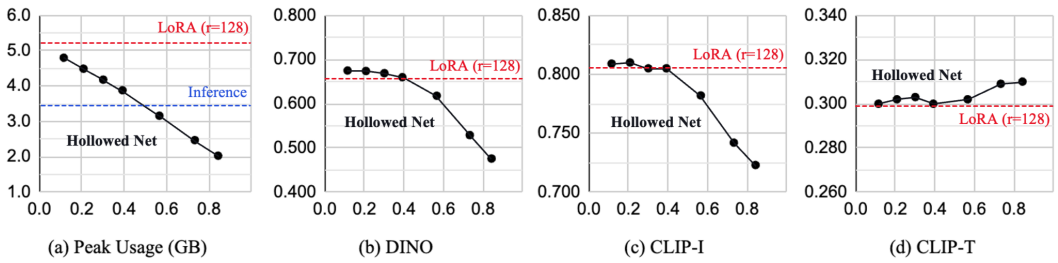


Figure 5: Analysis of different fractions of hollowed layers. For all figures, the x-axis represents the fractions of layers removed from the pre-trained diffusion U-Net. The y-axis corresponds to the metric used for each figure.

Quantitative Evaluation The quantitative results are displayed in Table 1. For comparison baselines, we implement full fine-tuning (Full FT) and LoRA fine-tuning (LoRA FT) methods with rank 128 and rank 1 [13]. We find that while Full FT results in slightly higher performance than other methods, particularly in terms of DINO, the differences between Full FT and Hollowed Net are not significant as it is within the range of standard deviations of Full FT results across different seeds. Moreover, Full FT requires more than 16GB of GPU memory which is nearly 4.7 times the memory cost of performing an inference with a diffusion U-Net. Clearly, this is not a feasible solution for on-device learning, where computation resources, especially memory I/O, are extremely limited.

Our Hollowed Net demonstrates its superior memory efficiency based on a significant reduction in model size, requiring only 3.88GB of GPU memory usage for fine-tuning. This is only an 11% increase compared to inference. Its personalization performance is comparable to or marginally better than that of LoRA fine-tuning using the same rank ($r = 128$), while LoRA requires a 50% increase in GPU memory compared to inference. Using the lowest rank of LoRA ($r = 1$) does not compete with Hollowed Net either, as its memory efficiency is limited by the need to run backpropagation on the entire U-Net, even though the number of fine-tuning parameters is significantly small. Additionally, the use of a low number of fine-tuning parameters significantly degrades personalization capacity.

For human evaluation, we conduct user studies with 40 participants, each completing a set of 25 comparative tasks. In each task, participants are presented with a reference image, a prompt, and two generated images (A and B). They answer two questions: subject fidelity and text fidelity. Each pair of generated images, A and B, is created using Hollowed Net and LoRA FT, and the labels (A or B) are randomly assigned for each task. Table 2 displays the results of these user studies. These findings confirm that users generally perceive the images generated by Hollowed Net and LoRA FT to be similar in both subject fidelity and text fidelity, consistent with the main results presented in Table 1.

Additionally, we include the analysis of computational loads for Hollowed Net and LoRA FT in Table 3. Each number corresponds to one step of each stage: one forward pass for pre-computing and inference and one forward+backward pass for fine-tuning. For the fine-tuning of Hollowed Net, $\sim$1000 steps are required, totaling $2.004 \times 1000 = 2004$ TFLOPs. For pre-computing, we find 200 pre-computed samples are sufficient to achieve high-fidelity results (see Appendix D for a detailed analysis), requiring $0.238 \times 200 = 47.6$ TFLOPs of additional computation. Therefore, the total computation required for training with Hollowed Net is $2004 + 47.6 = 2051.6$ TFLOPs, which is lower than $2.148 \times 1000 = 2148$ TFLOPs needed for LoRA FT. On the other hand, for running an inference pass, Hollowet Net requires approximately 0.204 TFLOPs more than LoRA FT, as it needs to repeat part of the early down-blocks to reproduce the path used in training.

Qualitative Evaluation In Fig. 4, we present qualitative generation results of Hollowed Net for various subjects and prompts. The upper half shows examples from the DreamBooth dataset, and the lower half displays examples from the CustomConcept101 dataset. These results demonstrate that Hollowed Net effectively captures the visual details of the target subjects, while maintaining high text-image alignment for different types of applications including property modification, recontextualization, accessorization, and artistic rendition. Its ability enabling high-fidelity personalization with memory costs as low as those of inference makes it an efficient solution for a range of on-device applications with constrained computational resources. Additional qualitative examples with SDXL [23] are included in Appendix E, illustrating the scalability of our approach in a larger model.

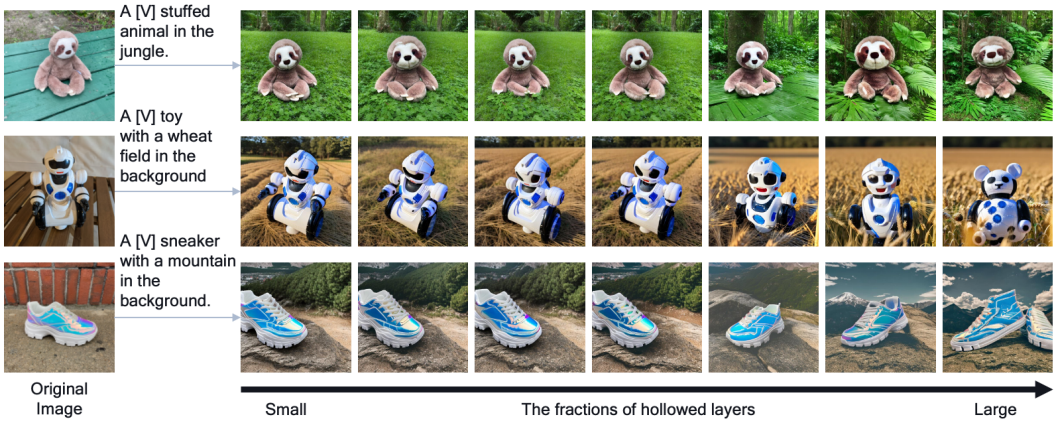


Figure 6: Qualitative results with different fractions of hollowed layers given three subjects from the DreamBooth dataset [5].

5.3 Ablation Study on Fractions of Hollowed Layers

Based on the symmetrical "U-shape" architecture of the diffusion U-Net, we can design different Hollowed Net architectures by selecting a different up-block layer skip-connected to a down-block layer and hollowing out the middle layers between the pair. Figure 5 presents experimental results across different fractions of hollowed layers, ranging from around 10% to 85% of layers removed. In Fig. 5, we observe the peak GPU memory usage decreasing linearly with layer removal, as fewer model weights need to be stored on the GPU during backpropagation. Analyzing the DINO and CLIP-I scores in Fig. 5 (b) and (c), we find that the model's capacity to preserve subject fidelity remains comparable to or slightly better than LoRA until around 39.2% of layers are removed, where memory cost reduces nearly to the level of inference. Beyond this threshold, however, subject fidelity significantly diminishes, as fewer layers essential for personalization are included in the Hollowed Net. This effect of hollowed layer fractions is also visible in the qualitative results in Fig. 6. Meanwhile, the CLIP-T score does not exhibit a general trend, except in cases of very high hollowed fractions, where the model is not capable of personalization, and thus generates images solely based on a given prompt. However, note that the increase in CLIP-T remains marginal, as Hollowed Nets with low hollowed fractions also maintain a high capacity for text-image alignment.

6 Conclusion

In conclusion, our paper introduces a novel approach for on-device personalization through memory-efficient fine-tuning with Hollowed Net. Hollowed Net effectively leverages the architecture of the diffusion U-Net, enabling fine-tuning with significantly reduced memory costs by minimizing the model's size during fine-tuning without requiring any additional processes such as structural pruning or pre-training on large-scale datasets. However, we observe that, due to the use of non-personalized prompts with the original network, the model's performance can be sensitive to the granularity of class token definitions. For example, the DreamBooth dataset contains "poop emoji" images, for which the class token is very coarsely defined as "toy". In this case, non-personalized intermediate activations generated with prompts using "toy" struggle to effectively correlate and generate "poop emoji" image. Therefore, a careful choice of fine-grained class tokens is necessary for the effective application of Hollowed Net.

Additionally, it is worth noting that our methodology is orthogonal to existing different PEFT methods [24, 25] and quantization methods [26, 27]. Thus, our approach offers substantial potential for further memory reduction, which is crucial for training under constrained computational resources. Furthermore, while our primary focus in this paper has been on image generation tasks, our method is not limited to diffusion models and can be seamlessly extended to various NLP tasks with LLMs, which we leave for future work. We anticipate that Hollowed Net will be applied to a wide range of tasks requiring constrained computational resources, serving as an efficient solution for various on-device applications.

References

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

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

[3] 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.

[4] 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.

[5] 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, licensed under CC BY 4.0.

[6] 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. *arXiv preprint arXiv:2311.13600*, 2023.

[7] 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.

[8] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Wei Wei, Tingbo Hou, Yael Pritch, Neal Wadhwa, Michael Rubinstein, and Kfir Aberman. Hyperdreambooth: Hypernetworks for fast personalization of text-to-image models. *arXiv preprint arXiv:2307.06949*, 2023.

[9] Rinon Gal, Moab Arar, Yuval Atzmon, Amit H Bermano, Gal Chechik, and Daniel Cohen-Or. Encoder-based domain tuning for fast personalization of text-to-image models. *ACM Transactions on Graphics (TOG)*, 42(4):1–13, 2023.

[10] Yuxiang Wei, Yabo Zhang, Zhilong Ji, Jinfeng Bai, Lei Zhang, and Wangmeng Zuo. Elite: Encoding visual concepts into textual embeddings for customized text-to-image generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15943–15953, 2023.

[11] Wenhu Chen, Hexiang Hu, Yandong Li, Nataniel Ruiz, Xuhui Jia, Ming-Wei Chang, and William W Cohen. Subject-driven text-to-image generation via apprenticeship learning. *Advances in Neural Information Processing Systems*, 36, 2024.

[12] Dongxu Li, Junnan Li, and Steven Hoi. Blip-diffusion: Pre-trained subject representation for controllable text-to-image generation and editing. *Advances in Neural Information Processing Systems*, 36, 2024.

[13] 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.

[14] Ligong Han, Yinxiao Li, Han Zhang, Peyman Milanfar, Dimitris Metaxas, and Feng Yang. Svdiff: Compact parameter space for diffusion fine-tuning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7323–7334, 2023.

[15] Jeffrey O Zhang, Alexander Sax, Amir Zamir, Leonidas Guibas, and Jitendra Malik. Side-tuning: a baseline for network adaptation via additive side networks. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, pages 698–714. Springer, 2020.

[16] Han Cai, Chuang Gan, Ligeng Zhu, and Song Han. Tinytl: Reduce activations, not trainable parameters for efficient on-device learning. *arXiv preprint arXiv:2007.11622*, 2020.

[17] Yizhe Zhang, Shubhankar Borse, Hong Cai, and Fatih Porikli. Auxadapt: Stable and efficient test-time adaptation for temporally consistent video semantic segmentation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2339–2348, 2022.

[18] Yi-Lin Sung, Jaemin Cho, and Mohit Bansal. Lst: Ladder side-tuning for parameter and memory efficient transfer learning. *Advances in Neural Information Processing Systems*, 35:12991–13005, 2022.

[19] Andrey Gromov, Kushal Tirumala, Hassan Shapourian, Paolo Glorioso, and Daniel A Roberts. The unreasonable ineffectiveness of the deeper layers. *arXiv preprint arXiv:2403.17887*, 2024.

[20] Bo-Kyeong Kim, Geonmin Kim, Tae-Ho Kim, Thibault Castells, Shinkook Choi, Junho Shin, and Hyoung-Kyu Song. Shortened llama: A simple depth pruning for large language models. *arXiv preprint arXiv:2402.02834*, 2024.

[21] Bo-Kyeong Kim, Hyoung-Kyu Song, Thibault Castells, and Shinkook Choi. Bk-sdm: A lightweight, fast, and cheap version of stable diffusion. *arXiv preprint arXiv:2305.15798*, 2023.

[22] Yijun Li, Richard Zhang, Jingwan Lu, and Eli Shechtman. Few-shot image generation with elastic weight consolidation. *arXiv preprint arXiv:2012.02780*, 2020.

[23] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023.

[24] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, 36, 2024.

[25] Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. Dora: Weight-decomposed low-rank adaptation. *arXiv preprint arXiv:2402.09353*, 2024.

[26] Yuzhang Shang, Zhihang Yuan, Bin Xie, Bingzhe Wu, and Yan Yan. Post-training quantization on diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1972–1981, 2023.

[27] Xiuyu Li, Yijiang Liu, Long Lian, Huanrui Yang, Zhen Dong, Daniel Kang, Shanghang Zhang, and Kurt Keutzer. Q-diffusion: Quantizing diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17535–17545, 2023.

A Experiments with Layer-Pruned Diffusion Models

As shown in recent work [21], layer pruning involves the complete removal of selected layers, which necessitates extensive pre-training on large datasets to recover lost information and restore model functionality. However, diffusion models often suffer from substantial performance degradation post-pruning, as the lost information may not be fully recoverable through pre-training.

Table 4 presents experiments with BK-SDM [21] models, layer-pruned SD models, using rank-128 LoRA. Compared to the results in Table 1, these models achieve memory usage comparable to Hollowed Net but show significant performance degradation. Despite extensive pre-training, their performance remains compromised.

In contrast, Hollowed Net does not completely remove deep layers and requires no additional pre-training. Instead, we temporarily exclude selected layers during fine-tuning while preserving essential information through a pre-computation stage. Notably, despite this added stage, the overall computational load for training Hollowed Net can remain more efficient than LoRA fine-tuning, as discussed in Sec. 5.2 and Appendix D.

Table 4: Quantitative results of BK-SDM on the DreamBooth dataset [5].

Method	# of Parameters	Training Memory	DINO	CLIP-I	CLIP-T
BK-SDM-Base	595.7M	3.546GB	0.629 ± 0.012	0.788 ± 0.007	0.300 ± 0.001
BK-SDM-Small	496.2M	3.133GB	0.602 ± 0.013	0.774 ± 0.008	0.298 ± 0.001

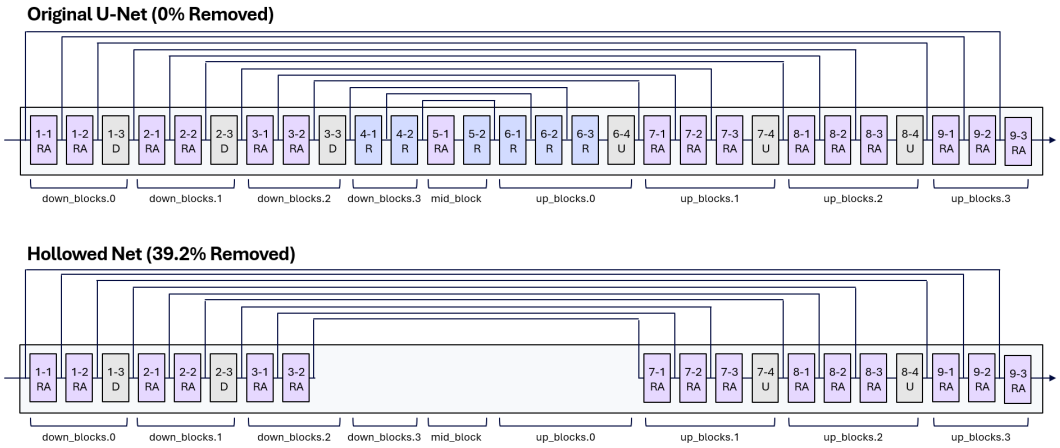


Figure 7: Architectural details of Hollowed Net. R represents ResBlock. RA represents a set of ResBlock and CrossAttentionBlock. D and U represent Downsample and Upsample Convs, respectively.

B Architectural Details of Hollowed Net

In this section, we present the architectural details of Hollowed Net. We leverage the skip connections inherent in the U-Net architecture to determine which layers to be removed during fine-tuning (hollowed). For our main results, we choose the third block of the down_blocks.2 (block 3-3), the entire down_blocks.3 (blocks 4-1 and 4-2), the entire mid_block (blocks 5-1 and 5-2), and the entire up_blocks.0 (blocks 6-1, 6-2, 6-3, and 6-4) to be hollowed, which corresponds to around 39.2% of the U-Net's parameters, as described in Fig. 7.

Similarly, Hollowed Net with different fractions of hollowed layers can be achieved as follows:

- 11.5% removed: blocks 5-1 and 5-2 are hollowed.
- 20.8% removed: blocks 4-2, 5-1, 5-2, and 6-1 are hollowed.

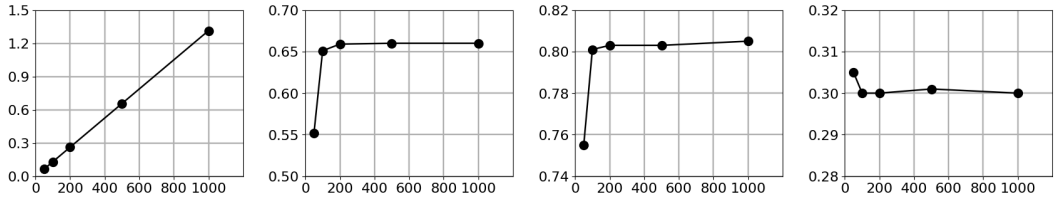
- 30.1% removed: blocks 4-1, 4-2, 5-1, 5-2, 6-1, and 6-2 are hollowed.
- 56.6% removed: blocks 3-2, 3-3, 4-1, 4-2, 5-1, 5-2, 6-1, 6-2, 6-3, 6-4, and 7-1 are hollowed.
- 73.3% removed: blocks 3-1, 3-2, 3-3, 4-1, 4-2, 5-1, 5-2, 6-1, 6-2, 6-3, 6-4, 7-1, and 7-2 are hollowed.
- 84.3% removed: blocks 2-3, 3-1, 3-2, 3-3, 4-1, 4-2, 5-1, 5-2, 6-1, 6-2, 6-3, 6-4, 7-1, 7-2, 7-3, and 7-4 are hollowed.

Table 5: Quantitative results of LoRA FT and Hollowed Net with different ranks

Method	# of Parameters	Training Memory	DINO	CLIP-I	CLIP-T
LoRA r=4	866.7M	4.847GB	0.564 ± 0.014	0.766 ± 0.006	0.311 ± 0.001
LoRA r=16	869.2M	4.883GB	0.618 ± 0.008	0.788 ± 0.005	0.305 ± 0.001
Hollow r=4	527.7M	3.526GB	0.566 ± 0.009	0.763 ± 0.003	0.311 ± 0.001
Hollow r=16	529.9M	3.558GB	0.626 ± 0.009	0.789 ± 0.005	0.305 ± 0.001

C Experiments with Different Ranks

In Table 5, we present the results using LoRA and Hollowed Net with different ranks (4 and 16) using the DreamBooth dataset. While the default rank of 4 in the diffusers library is often used, we have found that it often oversimplifies personalization details or fails to effectively handle a range of challenging subjects and prompts. Increasing the rank from 4 to 16 improves subject fidelity. However, to achieve personalization quality comparable to full fine-tuning across all subjects and prompts, we find that the rank of 128 is necessary.



(a) Data Storage (GB) (b) DINO (c) CLIP-I (d) CLIP-T

Figure 8: Analysis of different numbers of pre-computed samples. For all figures, the x-axis represents the number of pre-computed samples. The y-axis corresponds to the metric used for each figure.

Table 6: Resource usage analysis for different hollowed fractions and numbers of pre-computed samples

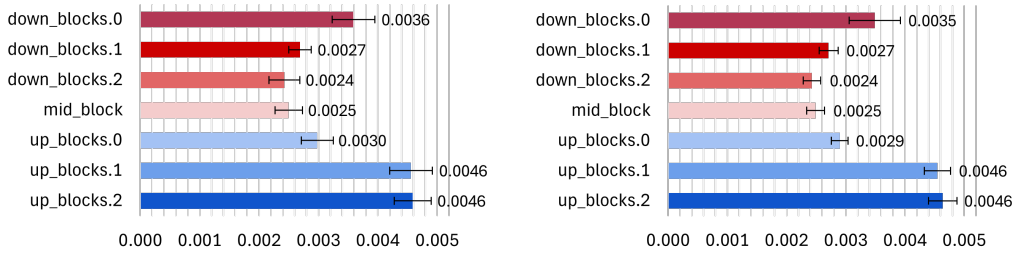
Hollowed Fraction	# of Precomputed Samples	Peak GPU Usage	Date Storage	Pre-computing FLOPs	Fine-tuning FLOPs	Total FLOPs
11.5%	200	4.49GB	0.08GB	44T	2081T	2125T
11.5%	500	4.49GB	0.19GB	109T	2081T	2190T
11.5%	1000	4.49GB	0.38GB	218T	2081T	2299T
39.2%	200	3.88GB	0.26GB	48T	2004T	2052T
39.2%	500	3.88GB	0.66GB	119T	2004T	2123T
39.2%	1000	3.88GB	1.31GB	238T	2004T	2242T
56.6%	200	3.16GB	0.26GB	56T	1776T	1832T
56.6%	500	3.16GB	0.66GB	140T	1776T	1916T
56.6%	1000	3.16GB	1.31GB	279T	1776T	2055T

D Further Analysis on Computational Costs

In Fig. 8, we provide ablation studies on the impact of varying the number of samples on both quantitative and qualitative results. The findings indicate that using only 200 pre-computed samples results in minimal performance degradation compared to using 1000 pre-computed samples.

Additionally, we present a detailed analysis of computational loads and space consumption in Table 6 for different numbers of precomputed samples and different fractions of hollowed layers, which

will enable users to choose the optimal configurations of Hollowed Net according to their specific resource constraints.



(a) DreamBooth (b) CustomConcept101

Figure 9: Analysis of the LoRA weight change before and after personalization, per block of U-Net using SDXL [23].

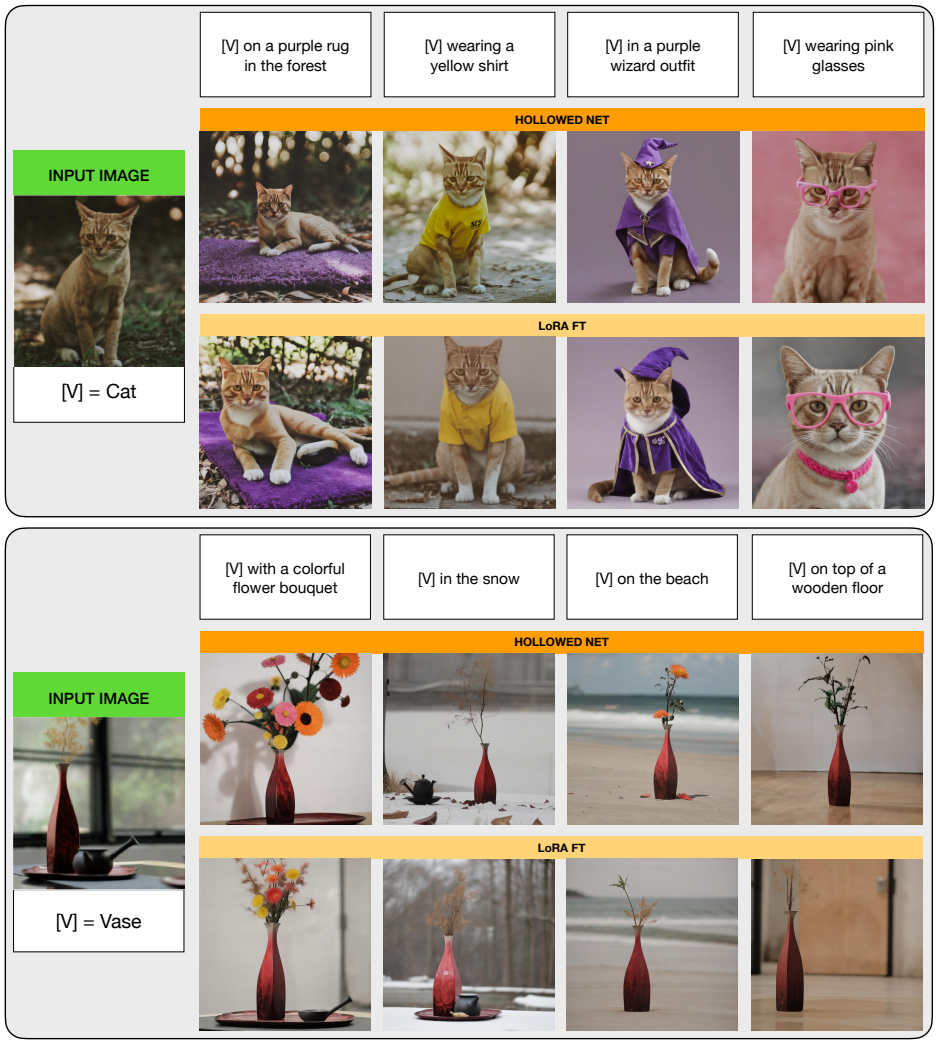


Figure 10: The qualitative examples of Hollowed Net and LoRA FT with the samples from the DreamBooth dataset [5] using SDXL [23].

E Experiments with SDXL

To demonstrate the scalability of Hollowed Net, we present additional analysis and qualitative examples using SDXL [23]. Figure 9 shows that similar patterns of weight changes are observable with SDXL, as displayed in Fig. 2. In Fig. 10, we present qualitative examples of Hollowed Net and LoRA FT with the samples from the DreamBooth dataset using SDXL. Hollowed Net is applied by removing the entire $\mathrm{mid\_block}$ layers (410M parameters) of SDXL. The results show that Hollowed Net achieves high-fidelity personalization results comparable to LoRA FT.

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: Refer Section 5 Experiments which includes the materials to support the answer.

   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: Refer Section 6 Conclusion which describes the limitation.

   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: [NA]

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: Refer Section 4 and 5 which describe the details of our architecture and experimental settings.

   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: [No]

Justification: We used a public dataset and described its experimental settings (Refer the Section 5). The code would be available after an internal review process has been completed (not available at this submission time).

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: Refer Section 5 Experiments which describes the details.

   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: Refer the table in Section 5 Experiments where the statistical variations of the results are included.

   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: Refer the Section 5 Experiments for the details.

   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: Conform with the 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: [NA]

    Justification: (based on the guideline described below) Our work does not consider to improve the quality of generative models itself but to optimize the network to fine-tune the model in a faster and more efficient way.

    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 work does not release any model or dataset but considers an efficient fine-tuning method.

    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: Our work uses the model and dataset with the license and terms of use explicitly mentioned in the release package.

    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: [NA]

    Justification: Our work does not release new assets.

    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: [NA]

    Justification: Our work does not involve crowdsourcing nor research.

    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: [NA]

    Justification: Our work does not involve crowdsourcing nor research.

    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.