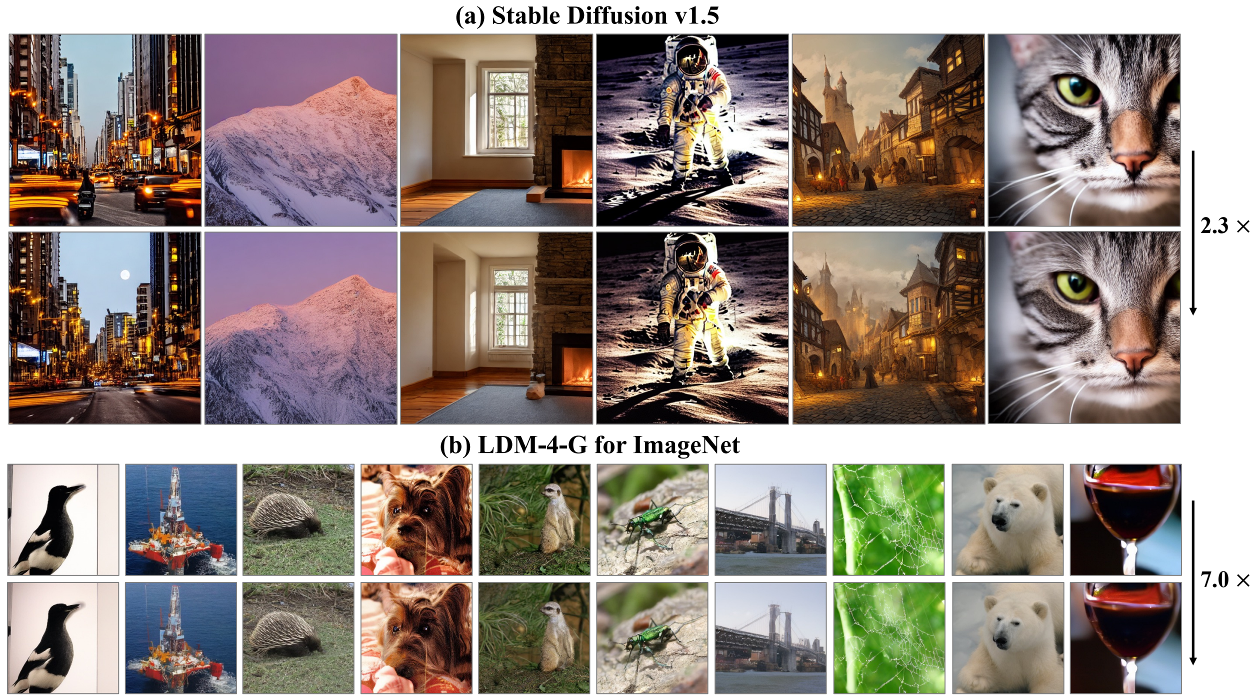
DeepCache: Accelerating Diffusion Models for Free

Xinyin Ma Gongfan Fang Xinchao Wang*
National University of Singapore
{maxinyin, gongfan}@u.nus.edu, xinchao@nus.edu.sg


(a) Stable Diffusion v1.5

(b) LDM-4-G for ImageNet

Figure 1. Accelerating Stable Diffusion V1.5 and LDM-4-G by 2.3× and 7.0×, with 50 PLMS steps and 250 DDIM steps respectively.

Abstract

Diffusion models have recently gained unprecedented attention in the field of image synthesis due to their remarkable generative capabilities. Notwithstanding their prowess, these models often incur substantial computational costs, primarily attributed to the sequential denoising process and cumbersome model size. Traditional methods for compressing diffusion models typically involve extensive retraining, presenting cost and feasibility challenges. In this paper, we introduce DeepCache, a novel training-free paradigm that accelerates diffusion models from the perspective of model architecture. DeepCache capitalizes on the inherent temporal redundancy observed in the sequential denoising steps of diffusion models, which caches and retrieves features across adjacent denoising stages, thereby curtailing redundant computations. Utilizing the property of the U-Net, we reuse the high-level features while updating the low-level features in a very cheap way. This innovative strategy, in turn, enables a speedup factor of 2.3× for Stable Diffusion v1.5 with only a 0.05 decline in CLIP Score, and 4.1× for LDM-4-G with a slight decrease of 0.22 in FID on ImageNet. Our experiments also demonstrate DeepCache's superiority over existing pruning and distillation methods that necessitate retraining and its compatibility with current sampling techniques. Furthermore, we find that under the same throughput, DeepCache effectively achieves comparable or even marginally improved results with DDIM or PLMS. Code is available at https://github.com/horseee/DeepCache.

1. Introduction

In recent years, diffusion models [9, 18, 59, 61] have emerged as a pivotal advancement in the field of genera-

* Corresponding author

tive modeling, gaining substantial attention for their impressive capabilities. These models have demonstrated remarkable efficacy across diverse applications, being employed for the generation of images [21, 62, 66], text [11, 30], audio [6, 46], and video [19, 38, 58]. A large number of attractive applications have been facilitated with diffusion models, including but not limited to image editing [2, 22, 40], image super-enhancing [28, 53], image-to-image translation [7, 51], text-to-image generation [43, 47, 48, 52] and text-to-3D generation [32, 37, 45].

Despite the significant effectiveness of diffusion models, their relatively slow inference speed remains a major obstacle to broader adoption, as highlighted in [31]. The core challenge stems from the step-by-step denoising process required during their reverse phase, limiting parallel decoding capabilities [57]. Efforts to accelerate these models have focused on two main strategies: reducing the number of sampling steps, as seen in approaches [36, 41, 54, 60], and decreasing the model inference overhead per step through methods like model pruning, distillation, and quantization [10, 13, 23].

Our goal is to enhance the efficiency of diffusion models by reducing model size at each step. Previous compression methods for diffusion models focused on re-designing network architectures through a comprehensive structural analysis [31] or involving frequency priors into the model design [68], which yields promising results on image generation. However, they require large-scale datasets for retraining these lightweight models. Pruning-based methods, as explored by [10, 23], lessen the data and training requirements to 0.1% of the full training. Alternatively, [34] employs adaptive models for different steps, which is also a potential solution. However, it depends on a collection of pre-trained models or requires optimization of subnetworks [67]. Those methods can reduce the expense of crafting a new lightweight model, but the retraining process is still inevitable, which makes the compression costly and less practical for large-scale pre-trained diffusion models, such as Stable Diffusion [49].

To this end, we focus on a challenging topic: *How to significantly reduce the computational overhead at each denoising step without additional training, thereby achieving a cost-free compression of Diffusion Models?* To achieve this, we turn our attention to the intrinsic characteristics of the reverse denoising process of diffusion models, observing a significant temporal consistency in the high-level features between consecutive steps. We found that those high-level features are even cacheable, which can be calculated once and retrieved again for the subsequent steps. By leveraging the structural property of U-Net, the high-level features can be cached while maintaining the low-level features updated at each denoising step. Through this, a considerable enhancement in the efficiency and speed of Diffusion Models can be achieved without any training.

To summarize, we introduce a novel paradigm for the acceleration of Diffusion Models, which gives a new perspective for training-free acceleration. It is not merely compatible with existing fast samplers but also shows potential for comparable or superior generation capabilities. The contributions of our paper include:

- We introduce a simple and effective acceleration algorithm, named DeepCache, for dynamically compressing diffusion models during runtime and thus enhancing image generation speed without additional training burdens.
- DeepCache utilizes the temporal consistency between high-level features. With the cacheable features, the redundant calculations are effectively reduced. Furthermore, we introduce a non-uniform 1:N strategy, specifically tailored for long caching intervals.
- DeepCache is validated across a variety of datasets, including CIFAR, LSUN-Bedroom/Churches, ImageNet, COCO2017 and PartiPrompt, and tested under DDPM, LDM, and Stable Diffusion. Experiments demonstrate that our approach has superior efficacy compared to pruning and distillation algorithms that require retraining under the same throughput.

2. Related Work

High-dimensional image generation has evolved significantly in generative modeling. Initially, GANs [1, 12] and VAEs [16, 24] led this field but faced scalability issues due to instability and mode collapse [25]. Recent advancements have been led by Diffusion Probabilistic Models [9, 18, 49, 63], which offer superior image generation. However, the inherent nature of the reverse diffusion process [61] slows down inference. Current research is focused on two main methods to speed up diffusion model inference.

Optimized Sampling Efficiency focuses on reducing the number of sampling steps. DDIM [60] reduces these steps by exploring a non-Markovian process, related to neural ODEs. Studies [3, 35, 36, 71] further dive into the fast solver of SDE or ODE to create efficient sampling of diffusion models. Some methods progressively distilled the model to reduced timestep [54] or replace the remaining steps with a single-step VAE [39]. The Consistency Model [64] converts random noise to the initial images with only one model evaluation. Parallel sampling techniques like DSNO [72] and ParaDiGMS [57] employ Fourier neural operators and Picard iterations for parallel decoding .

Optimized Structural Efficiency. This approach aims to reduce inference time at each sampling step. It leverages strategies like structural pruning in Diff-pruning [10] and efficient structure evolving in SnapFusion [31]. Spectral

Diffusion [68] enhances architectural design by incorporating frequency dynamics and priors. In contrast to these methods, which use a uniform model at each step, [34] proposes utilizing different models at various steps, selected from a diffusion model zoo. The early stopping mechanism in diffusion is explored in [29, 42, 65], while [13, 56] focus on low-precision weights and activations. Some other works [17, 20, 23] transferred the knowledge into the distilled model. Additionally, [4, 5] present novel approaches to concentrate on inputs, with the former adopting a unique forward process for each pixel and the latter merging tokens based on similarity in attention modules. Our method is categorized under an objective to minimize the average inference time per step. Uniquely, our approach reduces the average model size substantially for each step, accelerating the denoising process without necessitating retraining.

3. Methodology

3.1. Preliminary

Forward and Reverse Process. Diffusion models [18] simulate an image generation process using a series of random diffusion steps. The core idea behind diffusion models is to start from random noise and gradually refine it until it resembles a sample from the target distribution. In the forward diffusion process, with a data point sampled from the real distribution, $\mathbf{x}_0 \sim q(\mathbf{x})$, Gaussian noises are gradually added in T steps:

$$q\left(\mathbf{x}_t|\mathbf{x}_{t-1}\right) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}\right) \tag{1}$$

where t is the current timestep and $\{\beta_t\}$ schedules the noise. The *reverse diffusion process* denoises the random noise $x_T \sim \mathcal{N}(0, \mathbf{I})$ into the target distribution by modeling $q\left(\mathbf{x}_{t-1}|\mathbf{x}_t\right)$. At each reverse step t, the conditional probability distribution is approximated by a network $\epsilon_\theta\left(\mathbf{x}_t, t\right)$ with the timestep t and previous output $\mathbf{x}_t$ as input:

$$\begin{aligned} &x_{t-1} \sim p_\theta(x_{t-1}|x_t) = \\ &\mathcal{N}\left(x_{t-1}; \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta\left(\mathbf{x}_t, t\right)\right), \beta_t\mathbf{I}\right) \end{aligned} \tag{2}$$

where $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{i=1}^{T}\alpha_i$. Applied iteratively, it gradually reduces the noise of the current $\mathbf{x}_t$, bringing it close to a real data point when we reach x_0.

High-level and Low-level Features in U-Net. U-Net [50] was originally introduced for biomedical image segmentation and showcased a strong ability to amalgamate low-level and high-level features, attributed to the skip connections. U-Net is constructed on stacked downsampling and upsampling blocks, which encode the input image into a high-level representation and then decode it for downstream tasks. The block pairs, denoted as $\{D_i\}_{i=1}^d$ and $\{U_i\}_{i=1}^d$, are interconnected with additional skip paths. Those skip paths directly forward the rich and relatively more low-level information from D_i to U_i. During the forward propagation in the U-Net architecture, the data traverses concurrently through two pathways: the *main branch* and the *skip branch*. These branches converge at a concatenation module, with the *main branch* providing processed high-level feature from the preceding upsampling block U_{i+1}, and the *skip branch* contributing corresponding feature from the symmetrical block D_i. Therefore, at the heart of a U-Net model is a concatenation of low-level features from the skip branch, and the high-level features from the main branch, formalized as:

$$\text{Concat}(D_i(\cdot), U_{i+1}(\cdot)) \tag{3}$$

3.2. Feature Redundancy in Sequential Denoising

The inherent sequentiality of the denoising process in diffusion models presents a primary bottleneck in inference speed. Previous methods primarily employed strategies that involved skipping certain steps to address this issue. In this work, we revisit the entire denoising process, seeking to uncover specific properties that could be optimized to enhance inference efficiency.

Observation. *Adjacent steps in the denoising process exhibit significant temporal similarity in high-level features.*

In Figure 2, we provide empirical evidence related to this observation. The experiments elucidate two primary insights: 1) There is a noticeable temporal feature similarity between adjacent steps within the denoising process, indicating that the change between consecutive steps is typically minor; 2) Regardless of the diffusion model we used, for each timestep, at least 10% of the adjacent timesteps exhibit a high similarity (>0.95) to the current step, suggesting that certain high-level features change at a gradual pace. This phenomenon can be observed in a large number of well-established models like Stable Diffusion, LDM, and DDPM. In the case of DDPM for LSUN-Churches and LSUN-Bedroom, some timesteps even demonstrate a high degree of similarity to 80% of the other steps, as highlighted in the green line in Figure 2 (c).

Building upon these observations, our objective is to leverage this advantageous characteristic to accelerate the denoising process. Our analysis reveals that the computation often results in a feature remarkably similar to that of the previous step, thereby highlighting the existence of redundant calculations for optimization. We contend that allocating significant computational resources to regenerate these analogous feature maps constitutes an inefficiency. While incurring substantial computational expense, yields marginal benefits, it suggests a potential area for efficiency improvements in the speed of diffusion models.

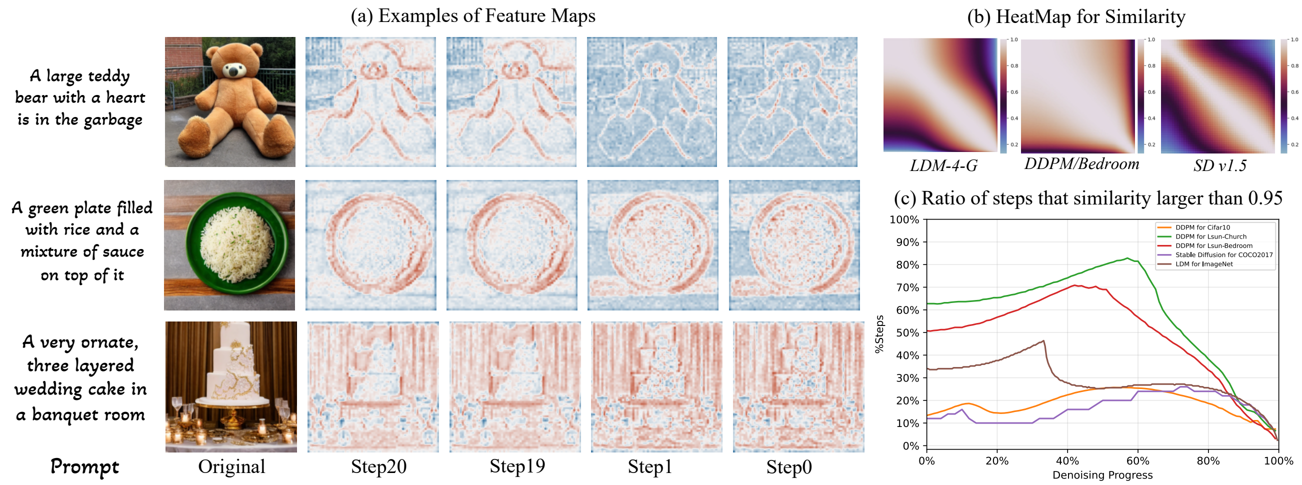


Figure 2. (a) Examples of feature maps in the up-sampling block U_2 in Stable Diffusion. We present a comparison from two adjacently paired steps, emphasizing the invariance inherent in the denoising process. (b) Heatmap of similarity between U_2's features in all steps on three typical diffusion models. (c) The percentage of steps with a similarity greater than 0.95 to the current step.

3.3. Deep Cache For Diffusion Models

We introduce DeepCache, a simple and effective approach that leverages the temporal redundancy between steps in the reverse diffusion process to accelerate inference. Our method draws inspiration from the caching mechanism in a computer system, incorporating a storage component designed for elements that exhibit minimal changes over time. Applying this in diffusion models, we eliminate redundant computations by strategically caching slowly evolving features, thereby obviating the need for repetitive recalculations in subsequent steps.

To achieve this, we shift our focus to the skip connections within U-Net, which inherently offers a dual-pathway advantage: the main branch requires heavy computation to traverse the entire network, while the skip branch only needs to go through some shallow layers, resulting in a very small computational load. The prominent feature similarity in the main branch allows us to reuse the already computed results rather than calculate it repeatedly for all timesteps.

Cacheable Features in denosing. To make this idea more concrete, we study the case within two consecutive timesteps t and $t-1$. According to the reverse process, x_{t-1} would be conditionally generated based on the previous results x_t. First, we generate x_t in the same way as usual, where the calculations are performed across the entire U-Net. To obtain the next output x_{t-1}, we retrieve the high-level features produced in the previous x_t. More specifically, consider a skip branch m in the U-Net, which bridges D_m and U_m, we cache the feature maps from the previous up-sampling block at the time t as the following:

$$F^t_{\text{cache}} \leftarrow U^t_{m+1}(\cdot) \tag{4}$$

which is the feature from the main branch at timestep t. Those cached features will be plugged into the network inference in the subsequent steps. In the next timestep $t-1$, the inference is not carried out on the entire network; instead, we perform a dynamic partial inference. Based on the previously generated x_t, we only calculate those that are necessary for the m-th skip branch and substitute the compute of the main branch with a retrieving operation from the cache in Equation 4. Therefore, the input for U^{t-1}_m in the $t-1$ timestep can be formulated as:

$$\text{Concat}(D^{t-1}_m(\cdot), F^t_{\text{cache}}) \tag{5}$$

Here, D^{t-1}_m represents the output of the m-th down-sampling block, which only contains a few layers if a small m is selected. For example, if we perform DeepCache at the first layer with $m = 1$, then we only need to execute one downsampling block to obtain D^{t-1}_1. As for the second feature F^t_{cache}, no additional computational cost is needed since it can be simply retrieved from the cache. We illustrate the above process in Figure 3.

Extending to 1:N Inference This process is not limited to the type with one step of full inference followed by one step of partial inference. As shown in Figure 2(b), pairwise similarity remains at a high value in several consecutive steps. The mechanism can be extended to cover more steps, with the cached features calculated once and reused in the consecutive $N-1$ steps to replace the original $U^{t-n}_{m+1}(\cdot)$, $n \in \{1, \dots, N-1\}$. Thus, for all the T steps for denoising, the sequence of timesteps that performs full inference are:

$$\mathcal{I} = \{x \in \mathbb{N} \,|\, x = iN,\ 0 \leq i < k\} \tag{6}$$

where $k = \lceil T/N \rceil$ denotes the times for cache updating.

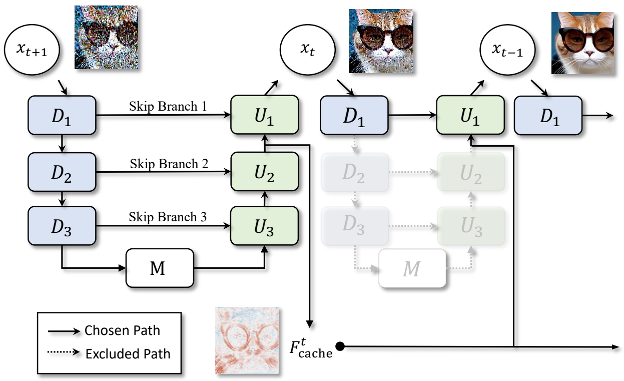


Figure 3. Illustration of DeepCache. At the $t-1$ step, x_{t-1} is generated by reusing the features cached at the t step, and the blocks D_2, D_3, U_2, U_3 are not executed for more efficient inference.

Non-uniform 1:N Inference Based on the 1:N strategy, we managed to accelerate the inference of diffusion under a strong assumption that the high-level features are invariant in the consecutive N step. However, it's not invariably the case, especially for a large N, as demonstrated by the experimental evidence in Figure 2(c). The similarity of the features does not remain constant across all steps. For models such as LDM, the temporal similarity of features would significantly decrease around 40% of the denoising process. Thus, for the non-uniform 1:N inference, we tend to sample more on those steps with relatively small similarities to the adjacent steps. Here, the sequence of timesteps to perform full inference becomes:

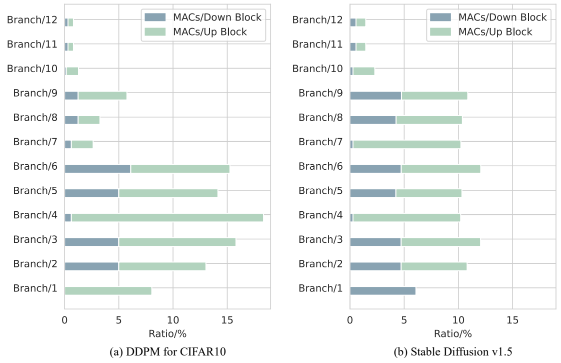
$$\mathcal{L} = \left\{ l_i \mid l_i \in \text{linear\_space}\left((-c)^{\frac{1}{p}}, (T-c)^{\frac{1}{p}}, k\right)\right\} \tag{7}$$
$$\mathcal{I} = \text{unique\_int}\left(\{i_k \mid i_k = (l_k)^p + c, \text{ where } l_k \in \mathcal{L}\}\right)$$

where $\text{linear\_space}(s, e, n)$ evenly spaces n numbers from s to e (exclusive) and $\text{unique\_int}(\cdot)$ convert the number to int and ensure the uniqueness of timesteps in the sequence. c is the hyper-parameter for the selected center timestep. In this equation, the frequency of indexes decreases as it moves away from a central timestep. It is essential to note that the aforementioned strategy represents one among several potential strategies. Alternative sequences, particularly centered on a specific timestep, can also yield similar improvements in image quality.

4. Experiment

4.1. Experimental Settings

Models, Datasets and Evaluation Metrics To demonstrate the effectiveness of our method is agnostic with the type of pre-trained DMs, we evaluate our method on three commonly used DMs: DDPM [18], LDM [49] and Stable Diffusion [49][1]. Except for this, to show that our method is compatible with the fast sampling methods, we build our method upon 100-step DDIM [60] for DDPM, 250-step for LDM and 50-step PLMS [35] for Stable Diffusion, instead of the complete 1000-step denoising process. We select six datasets that are commonly adopted to evaluate these models, including CIFAR10 [26], LSUN-Bedroom [69], LSUN-Churches [69], ImageNet [8], MS-COCO 2017 [33] and PartiPrompts [70]. For MS-COCO 2017 and PartiPrompt, we utilized the 5k validation set and 1.63k captions respectively as prompts for Stable Diffusion. For other datasets, we generate 50k images to assess the generation quality. We follow previous works [10, 57, 68] to employ the evaluation metrics including FID, sFID, IS, Precision-and-Recall and CLIP Score (on ViT-g/14) [14, 15, 27, 55].



(a) DDPM for CIFAR10

(b) Stable Diffusion v1.5

Figure 4. MACs for each skip branch, evaluated on DDPM for CIFAR10 and Stable Diffusion V1.5.

Baselines We choose Diff-Pruning [10] as the main baseline for our method since Diff-Pruning also reduces the training effort for compressing DMs. For the experiment on LDM, we extend [68] as another baseline to represent one of the best lightweight diffusion models. For the experiment on Stable Diffusion, we choose BK-SDMs [23], which are trained on 2.3M LAION image-text pairs, as baselines of architectural compression and distillation.

4.2. Complexity Analysis

We first analyze the improvement in inference speed facilitated by DeepCache. The notable acceleration in inference speed primarily arises from incomplete reasoning in denoising steps, with layer removal accomplished by partitioning the U-Net by the skip connection. In Figure 4, we present the division of multiply-accumulate operations (MACs) on two models. For each skip branch i, the MACs here contain the MACs in down block D_i and the up block U_i. There is a difference in the amount of computation allocated to different skip branches for different models. Stable diffusion

[1] https://huggingface.co/runwayml/stable-diffusion-v1-5

	ImageNet 256 × 256								
Method	**MACs ↓**	**Throughput ↑**	**Speed ↑**	**Retrain**	**FID ↓**	**sFID ↓**	**IS ↑**	**Precision ↑**	**Recall ↑**
IDDPM [44]	1416.3G	-	-	✗	12.26	5.42	-	70.0	62.0
ADM-G [9]	1186.4G	-	-	✗	4.59	5.25	186.70	82.0	52.0
LDM-4 [49]	99.82G	0.178	1×	✗	3.60	-	247.67	87.0	48.0
LDM-4*	99.82G	0.178	1×	✗	3.37	5.14	204.56	82.71	53.86
Spectral DPM [68]	9.9G	-	-	✓	10.60	-	-	-	-
Diff-Pruning [10]*	52.71G	0.269	1.51×	✓	$9.27_{(9.16)}$	10.59	$214.42_{(201.81)}$	87.87	30.87
Uniform - N=2	52.12G	0.334	1.88×	✗	3.39	5.11	204.09	82.75	54.07
Uniform - N=3	36.48G	0.471	2.65×	✗	3.44	5.11	202.79	82.65	53.81
Uniform - N=5	23.50G	0.733	4.12×	✗	3.59	5.16	200.45	82.36	53.31
Uniform - N=10	13.97G	1.239	6.96×	✗	4.41	5.57	191.11	81.26	51.53
Uniform - N=20	9.39G	1.876	10.54×	✗	8.23	8.08	161.83	75.31	50.57
NonUniform - N=10	13.97G	1.239	6.96×	✗	4.27	5.42	193.11	81.75	51.84
NonUniform - N=20	9.39G	1.876	10.54×	✗	7.11	7.34	167.85	77.44	50.08

Table 1. Class-conditional generation quality on ImageNet using LDM-4-G. The baselines here, as well as our methods, employ 250 DDIM steps. *We reproduce Diff-Pruning to have a comprehensive comparison and the official results are shown in brackets.

	CIFAR-10 32 × 32				
Method	**MACs ↓**	**Throughput ↑**	**Speed ↑**	**Retrain Steps ↓**	**FID ↓**
DDPM	6.1G	9.79	1×	-	4.19
DDPM*	6.1G	9.79	1×	-	4.16
Diff-Pruning	3.4G	13.45	1.37×	100k	5.29
Ours - N=2	4.15G	13.73	1.40×	0	4.35
Ours - N=3	3.54G	15.74	1.61×	0	4.70
Ours - N=5	3.01G	18.11	1.85×	0	5.73
Ours - N=10	2.63G	20.26	2.07×	0	9.74
	LSUN-Bedroom 256 × 256				
Method	**MACs ↓**	**Throughput ↑**	**Speed ↑**	**Retrain Steps ↓**	**FID ↓**
DDPM	248.7G	0.21	1×	-	6.62
DDPM*	248.7G	0.21	1×	-	6.70
Diff-Pruning	138.8G	0.31	1.48×	200k	18.60
Ours - N=2	190.8G	0.27	1.29×	0	6.69
Ours - N=3	172.3G	0.30	1.43×	0	7.20
Ours - N=5	156.0G	0.31	1.48×	0	9.49
	LSUN-Churches 256 × 256				
Method	**MACs ↓**	**Throughput ↑**	**Speed ↑**	**Retrain Steps ↓**	**FID ↓**
DDPM	248.7G	0.21	1×	-	10.58
DDPM*	248.7G	0.21	1×	-	10.87
Diff-Pruning	138.8G	0.31	1.48×	500k	13.90
Ours - N=2	190.8G	0.27	1.29×	0	11.31
Ours - N=3	172.3G	0.30	1.43×	0	11.75
Ours - N=5	156.0G	0.31	1.48×	0	13.68

Table 2. Results on CIFAR-10, LSUN-Bedroom and LSUN-Churches. All the methods here adopt 100 DDIM steps. * means the reproduced results, which are more comparable with our results since the random seed is the same.

demonstrates a comparatively uniform distribution across layers, whereas DDPM exhibits more computational burden concentrated within the first several layers. Our approach would benefit from U-Net structures that have a larger number of skip branches, facilitating finer divisions of models, and giving us more choices for trade-off the speed and quality. In our experiment, we choose the skip branch 3/1/2 for DDPMs, LDM-4-G and Stable Diffusion respectively. We provide the results of using different branches in Appendix.

To comprehensively evaluate the efficiency of our method, in the following experiments, we report the throughput of each model using a single RTX2080 GPU. Besides, we report MACs in those tables, which refer to the average MACs for all steps.

4.3. Comparison with Compression Methods

LDM-4-G for ImageNet. We conduct experiments on ImageNet, and the results are shown in Table 1. When accelerating to 4.1× the speed, a minor performance decline is observed (from 3.39 to 3.59). Compared with the pruning and distillation methods, a notable improvement over those methods is observed in FID and sFID, even in cases when the acceleration ratio of our method is more substantial. Furthermore, the augmentation in quality becomes more obvious with a larger number N of caching intervals if we employ the non-uniform 1:N strategy. Detailed results and hyper-parameters for the non-uniform 1:N strategy with small N are provided in the Appendix.

DDPMs for CIFAR-10 and LSUN. The results on CIFAR10, LSUN-Bedroom and LSUN-Churches are shown in Table 2. From these tables, we can find out that our method surpasses those requiring retraining, even though our methods have no retraining cost. Additionally, since we adopt a layer-pruning approach, which is more hardware-friendly, our acceleration ratio is more significant compared to the baseline method, under similar MACs constraints.

Stable Diffusion. The results are presented in Table 3. We outperform all three variants of BK-SDM, even with a faster denoising speed. As evident in Figure 5, the images generated by our method exhibit a greater consistency with the images generated by the original diffusion model, and

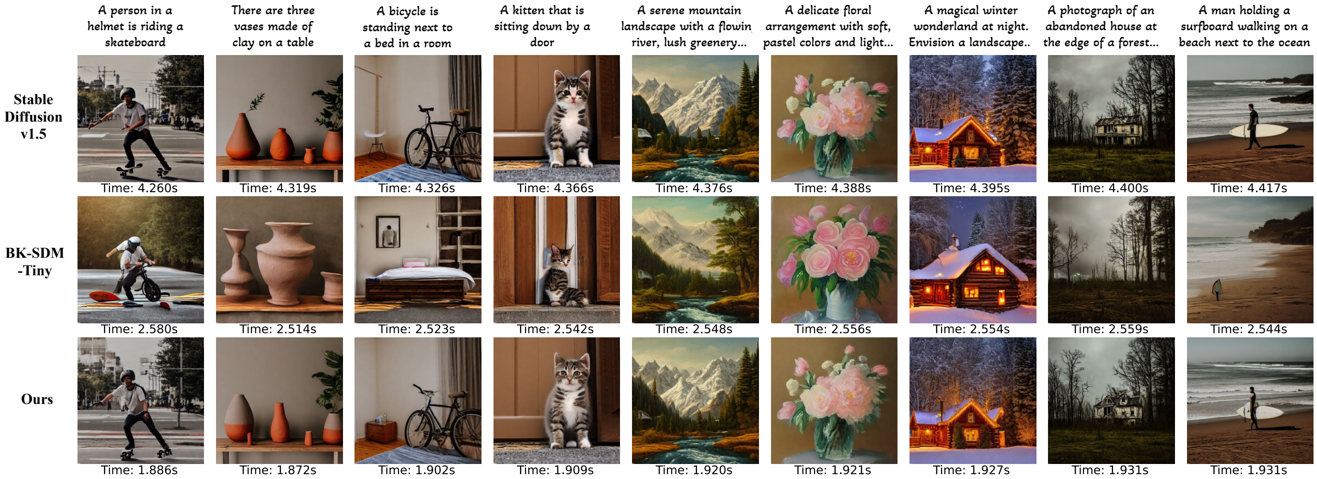

A person in a helmet is riding a skateboard
There are three vases made of clay on a table
A bicycle is standing next to a bed in a room
A kitten that is sitting down by a door
A serene mountain landscape with a flowin river, lush greenery...
A delicate floral arrangement with soft, pastel colors and light...
A magical winter wonderland at night. Envision a landscape..
A photograph of an abandoned house at the edge of a forest...
A man holding a surfboard walking on a beach next to the ocean
Stable Diffusion v1.5
Time: 4.260s
Time: 4.319s
Time: 4.326s
Time: 4.366s
Time: 4.376s
Time: 4.388s
Time: 4.395s
Time: 4.400s
Time: 4.417s
BK-SDM -Tiny
Time: 2.580s
Time: 2.514s
Time: 2.523s
Time: 2.542s
Time: 2.548s
Time: 2.556s
Time: 2.554s
Time: 2.559s
Time: 2.544s
Ours
Time: 1.886s
Time: 1.872s
Time: 1.902s
Time: 1.909s
Time: 1.920s
Time: 1.921s
Time: 1.927s
Time: 1.931s
Time: 1.931s

Figure 5. Visualization of the generated images by BK-SDM-Tiny and DeepCache. All the methods adopt the 50-step PLMS. The time here is the duration to generate a single image. Some prompts are omitted from this section for brevity. See Appendix for details.

	PartiPrompts				COCO2017			
Method	**MACs ↓**	**Throughput ↑**	**Speed ↑**	**CLIP Score ↑**	**MACs ↓**	**Throughput ↑**	**Speed ↑**	**CLIP Score ↑**
PLMS - 50 steps	338.83G	0.230	1.00×	29.51	338.83G	0.237	1.00×	30.30
PLMS - 25 steps	169.42G	0.470	2.04×	29.33	169.42G	0.453	1.91×	29.99
BK-SDM - Base	223.81G	0.343	1.49×	28.88	223.81G	0.344	1.45 ×	29.47
BK-SDM - Small	217.78G	0.402	1.75×	27.94	217.78G	0.397	1.68×	27.96
BK-SDM - Tiny	205.09G	0.416	1.81×	27.36	205.09G	0.415	1.76 ×	27.41
Ours	130.45G	0.494	2.15×	29.46	130.45G	0.500	2.11×	30.23

Table 3. Comparison with PLMS and BK-SDM. We utilized prompts in PartiPrompts and COCO2017 validation set to generate images at the resolution of 512. We choose N=5 to achieve a throughput that is comparable to or surpasses that of established baseline methods.

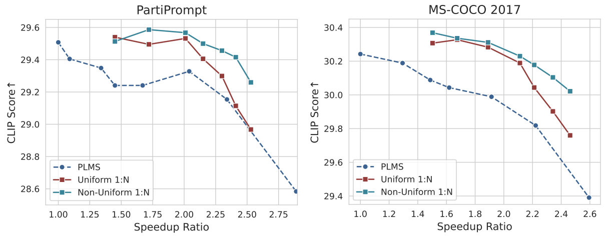


Figure 6. Comparison between PLMS, DeepCache with uniform 1:N and non-uniform 1:N strategies.

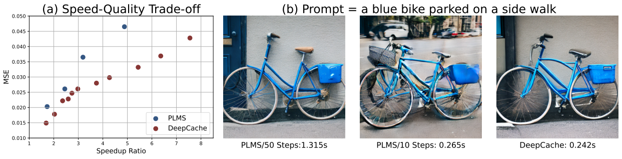


Figure 7. (a) MSE between images under 50-step PLMS and accelerating algorithms. (b) Qualitative Comparison. Here we build DeepCache upon the 20-step PLMS with the interval set to 3.

the image aligns better with the textual prompt. Results for other choices of N can be found in Figure 6.

Method	**Throughput ↑**	**FID ↓**	**sFID ↓**
DDIM - 59 steps	0.727	**3.59**	**5.14**
Ours	0.733	**3.59**	5.16
DDIM - 91 steps	0.436	3.46	**5.06**
Ours	0.471	**3.44**	5.11

Table 4. Comparison with DDIM under the same throughput. Here we conduct class-conditional ImageNet using LDM-4-G.

4.4. Comparison with Fast Sampler.

We conducted a comparative analysis with methods focused on reducing sampling steps. In Table 3, Table 4 and Figure 6, we compared our method with PLMS [35] and DDIM [60] under similar throughputs by increasing the interval N or reducing the timesteps correspondingly. Our method achieved slightly better results than the 25-step PLMS on Stable Diffusion. Alternatively, we can combine PLMS with DeepCache to achieve better performance. In Figure 7, we incorporate DeepCache alongside PLMS, resulting in improved consistency with the original images at a faster speed compared to solely utilizing PLMS. More comparisons can be found in the Appendix.

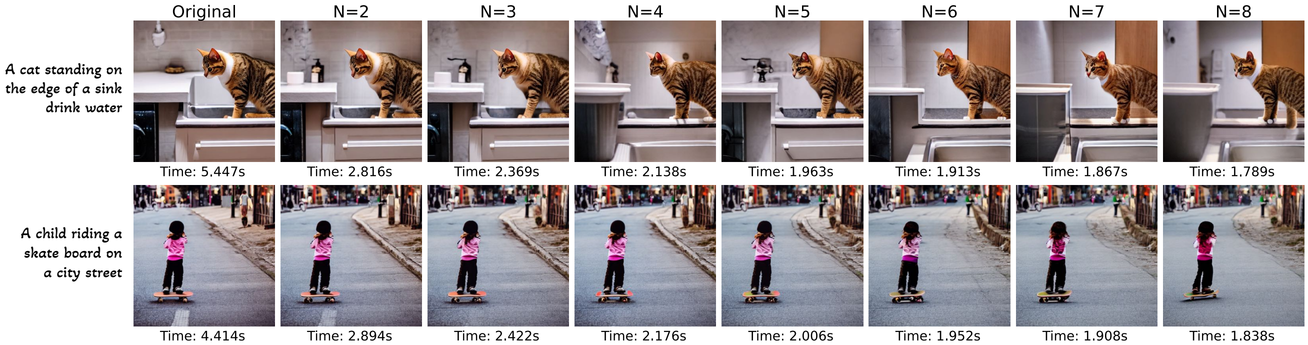


Figure 8. Illustration of the evolution in generated images with increasing caching interval N.

4.5. Analysis

Ablation Study. DeepCache can be conceptualized as incorporating $(N-1)\times K$ steps of shallow network inference on top of the DDIM's K steps, along with more updates of the noisy images. It is important to validate whether the additional computations of shallow network inference and the caching of features yield positive effectiveness: 1) **Effectiveness of Cached Features:** We assess the impact of cached features in Table 5. Remarkably, we observe that, without any retraining, the cached features play a pivotal role in the effective denoising of diffusion models employing a shallow U-Net. 2) **Positive Impact of Shallow Network Inference:** Building upon the cached features, the shallow network inference we conduct has a positive impact compared to DDIM. Results presented in Table 6 indicate that, with the additional computation of the shallow U-Net, DeepCache improves the 50-step DDIM by 0.32 and the 10-step DDIM by 2.98.

Illustration of the increasing caching interval N. In Figure 8, we illustrate the evolution of generated images as we increment the caching interval. A discernible trend emerges as a gradual reduction in time, revealing that the primary features of the images remain consistent with their predecessors. However, subtle details such as the color of clothing and the shape of the cat undergo modifications. Quantitative insights are provided in Table 1 and Figure 6, where with an interval $N<5$, there is only a slight reduction in the quality of the generated image.

5. Limitations

The primary limitation of our method originates from its dependence on the pre-defined structure of the pre-trained diffusion model. Specifically, when a model's shallowest skip branch encompasses a substantial portion of computation, such as 50% of the whole model, the achievable speedup ratio through our approach becomes relatively constrained. Additionally, our method encounters non-negligible performance degradation with larger caching steps (e.g., N=20), which could impose constraints on the acceleration ratio.

Model	Dataset	DeepCache	w/o Cached Features
DDPM	Cifar10	9.74	192.98
LDM-4-G	ImageNet	7.36	312.12

Table 5. Effectiveness of Cached Features. Under identical hyper-parameters, we replace the cached features with a zero matrix.

Steps	DDIM FID↓	DeepCache FID↓	Δ
50	4.67	4.35	-0.32
20	6.84	5.73	-1.11
10	13.36	10.38	-2.98

Table 6. Effectiveness of Shallow Network Inference on CIFAR-10. Steps here mean the number of steps that perform full model inference.

6. Conclusion

In this paper, we introduce a novel paradigm, DeepCache, to accelerate the diffusion model. Our strategy employs the similarity observed in high-level features across adjacent steps of the diffusion model and leverages the structural attributes in the U-Net architecture to facilitate the updating of low-level features. Empirical evaluations on several datasets and diffusion models demonstrate that DeepCache surpasses other compression methods that focus on the reduction of parameter size. Moreover, the proposed algorithm demonstrates comparable and even slightly superior generation quality compared to DDIM and PLMS, thereby offering a new perspective in the field.

Acknowledgement

This project is supported by the National Research Foundation Singapore under its Medium Sized Center for Advanced Robotics Technology Innovation, and the Singapore Ministry of Education Academic Research Fund Tier 1 (WBS: A-0009440-01-00).

References

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017. 2

[2] Omri Avrahami, Dani Lischinski, and Ohad Fried. Blended diffusion for text-driven editing of natural images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18208–18218, 2022. 2

[3] Fan Bao, Chongxuan Li, Jun Zhu, and Bo Zhang. Analytic-dpm: an analytic estimate of the optimal reverse variance in diffusion probabilistic models. *arXiv preprint arXiv:2201.06503*, 2022. 2

[4] Georgios Batzolis, Jan Stanczuk, Carola-Bibiane Schönlieb, and Christian Etmann. Non-uniform diffusion models. *arXiv preprint arXiv:2207.09786*, 2022. 3

[5] Daniel Bolya and Judy Hoffman. Token merging for fast stable diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4598–4602, 2023. 3

[6] Nanxin Chen, Yu Zhang, Heiga Zen, Ron J Weiss, Mohammad Norouzi, and William Chan. Wavegrad: Estimating gradients for waveform generation. *arXiv preprint arXiv:2009.00713*, 2020. 2

[7] Jooyoung Choi, Sungwon Kim, Yonghyun Jeong, Youngjune Gwon, and Sungroh Yoon. Ilvr: Conditioning method for denoising diffusion probabilistic models. *arXiv preprint arXiv:2108.02938*, 2021. 2

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 5

[9] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021. 1, 2, 6

[10] Gongfan Fang, Xinyin Ma, and Xinchao Wang. Structural pruning for diffusion models. In *Advances in Neural Information Processing Systems*, 2023. 2, 5, 6

[11] Shansan Gong, Mukai Li, Jiangtao Feng, Zhiyong Wu, and LingPeng Kong. Diffuseq: Sequence to sequence text generation with diffusion models. *arXiv preprint arXiv:2210.08933*, 2022. 2

[12] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014. 2

[13] Yefei He, Luping Liu, Jing Liu, Weijia Wu, Hong Zhou, and Bohan Zhuang. Ptqd: Accurate post-training quantization for diffusion models. *arXiv preprint arXiv:2305.10657*, 2023. 2, 3

[14] Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. Clipscore: A reference-free evaluation metric for image captioning. *arXiv preprint arXiv:2104.08718*, 2021. 5

[15] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017. 5

[16] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *International conference on learning representations*, 2016. 2

[17] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 3

[18] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 1, 2, 3, 5

[19] Jonathan Ho, William Chan, Chitwan Saharia, Jay Whang, Ruiqi Gao, Alexey Gritsenko, Diederik P Kingma, Ben Poole, Mohammad Norouzi, David J Fleet, et al. Imagen video: High definition video generation with diffusion models. *arXiv preprint arXiv:2210.02303*, 2022. 2

[20] Yongcheng Jing, Yiding Yang, Xinchao Wang, Mingli Song, and Dacheng Tao. Amalgamating knowledge from heterogeneous graph neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15709–15718, 2021. 3

[21] Bahjat Kawar, Michael Elad, Stefano Ermon, and Jiaming Song. Denoising diffusion restoration models. *Advances in Neural Information Processing Systems*, 35:23593–23606, 2022. 2

[22] Bahjat Kawar, Shiran Zada, Oran Lang, Omer Tov, Huiwen Chang, Tali Dekel, Inbar Mosseri, and Michal Irani. Imagic: Text-based real image editing with diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6007–6017, 2023. 2

[23] Bo-Kyeong Kim, Hyoung-Kyu Song, Thibault Castells, and Shinkook Choi. On architectural compression of text-to-image diffusion models. *arXiv preprint arXiv:2305.15798*, 2023. 2, 3, 5

[24] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 2

[25] Naveen Kodali, Jacob Abernethy, James Hays, and Zsolt Kira. On convergence and stability of gans. *arXiv preprint arXiv:1705.07215*, 2017. 2

[26] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 5

[27] Tuomas Kynkäänniemi, Tero Karras, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Improved precision and recall metric for assessing generative models. *Advances in Neural Information Processing Systems*, 32, 2019. 5

[28] Haoying Li, Yifan Yang, Meng Chang, Shiqi Chen, Huajun Feng, Zhihai Xu, Qi Li, and Yueting Chen. Srdiff: Single image super-resolution with diffusion probabilistic models. *Neurocomputing*, 479:47–59, 2022. 2

[29] Lijiang Li, Huixia Li, Xiawu Zheng, Jie Wu, Xuefeng Xiao, Rui Wang, Min Zheng, Xin Pan, Fei Chao, and Rongrong Ji. Autodiffusion: Training-free optimization of time steps and architectures for automated diffusion model acceleration. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7105–7114, 2023. 3

[30] Xiang Li, John Thickstun, Ishaan Gulrajani, Percy S Liang, and Tatsunori B Hashimoto. Diffusion-lm improves controllable text generation. *Advances in Neural Information Processing Systems*, 35:4328–4343, 2022. 2

[31] Yanyu Li, Huan Wang, Qing Jin, Ju Hu, Pavlo Chemerys, Yun Fu, Yanzhi Wang, Sergey Tulyakov, and Jian Ren. Snapfusion: Text-to-image diffusion model on mobile devices within two seconds. *arXiv preprint arXiv:2306.00980*, 2023. 2

[32] Chen-Hsuan Lin, Jun Gao, Luming Tang, Towaki Takikawa, Xiaohui Zeng, Xun Huang, Karsten Kreis, Sanja Fidler, Ming-Yu Liu, and Tsung-Yi Lin. Magic3d: High-resolution text-to-3d content creation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 300–309, 2023. 2

[33] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014. 5

[34] Enshu Liu, Xuefei Ning, Zinan Lin, Huazhong Yang, and Yu Wang. Oms-dpm: Optimizing the model schedule for diffusion probabilistic models. *arXiv preprint arXiv:2306.08860*, 2023. 2, 3

[35] Luping Liu, Yi Ren, Zhijie Lin, and Zhou Zhao. Pseudo numerical methods for diffusion models on manifolds. *arXiv preprint arXiv:2202.09778*, 2022. 2, 5, 7

[36] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *Advances in Neural Information Processing Systems*, 35:5775–5787, 2022. 2

[37] Shitong Luo and Wei Hu. Diffusion probabilistic models for 3d point cloud generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2837–2845, 2021. 2

[38] Zhengxiong Luo, Dayou Chen, Yingya Zhang, Yan Huang, Liang Wang, Yujun Shen, Deli Zhao, Jingren Zhou, and Tieniu Tan. Videofusion: Decomposed diffusion models for high-quality video generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10209–10218, 2023. 2

[39] Zhaoyang Lyu, Xudong Xu, Ceyuan Yang, Dahua Lin, and Bo Dai. Accelerating diffusion models via early stop of the diffusion process. *arXiv preprint arXiv:2205.12524*, 2022. 2

[40] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations. *arXiv preprint arXiv:2108.01073*, 2021. 2

[41] Chenlin Meng, Robin Rombach, Ruiqi Gao, Diederik Kingma, Stefano Ermon, Jonathan Ho, and Tim Salimans. On distillation of guided diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14297–14306, 2023. 2

[42] Taehong Moon, Moonseok Choi, EungGu Yun, Jongmin Yoon, Gayoung Lee, and Juho Lee. Early exiting for accelerated inference in diffusion models. In *ICML 2023 Workshop on Structured Probabilistic Inference $\{\backslash\&\}$ Generative Modeling*, 2023. 3

[43] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021. 2

[44] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pages 8162–8171. PMLR, 2021. 6

[45] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*, 2022. 2

[46] Vadim Popov, Ivan Vovk, Vladimir Gogoryan, Tasnima Sadekova, and Mikhail Kudinov. Grad-tts: A diffusion probabilistic model for text-to-speech. In *International Conference on Machine Learning*, pages 8599–8608. PMLR, 2021. 2

[47] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1 (2):3, 2022. 2

[48] 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. 2

[49] 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. 2, 5, 6

[50] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pages 234–241. Springer, 2015. 3

[51] Chitwan Saharia, William Chan, Huiwen Chang, Chris Lee, Jonathan Ho, Tim Salimans, David Fleet, and Mohammad Norouzi. Palette: Image-to-image diffusion models. In *ACM SIGGRAPH 2022 Conference Proceedings*, pages 1–10, 2022. 2

[52] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in Neural Information Processing Systems*, 35:36479–36494, 2022. 2

[53] Chitwan Saharia, Jonathan Ho, William Chan, Tim Salimans, David J Fleet, and Mohammad Norouzi. Image super-resolution via iterative refinement. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(4):4713–4726, 2022. 2

[54] Tim Salimans and Jonathan Ho. Progressive distillation

for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022. 2

[55] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *Advances in neural information processing systems*, 29, 2016. 5

[56] 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. 3

[57] Andy Shih, Suneel Belkhale, Stefano Ermon, Dorsa Sadigh, and Nima Anari. Parallel sampling of diffusion models. *arXiv preprint arXiv:2305.16317*, 2023. 2, 5

[58] Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, et al. Make-a-video: Text-to-video generation without text-video data. *arXiv preprint arXiv:2209.14792*, 2022. 2

[59] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR, 2015. 1

[60] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020. 2, 5, 7

[61] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019. 1, 2

[62] Yang Song, Sahaj Garg, Jiaxin Shi, and Stefano Ermon. Sliced score matching: A scalable approach to density and score estimation. In *Uncertainty in Artificial Intelligence*, pages 574–584. PMLR, 2020. 2

[63] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020. 2

[64] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. 2023. 2

[65] Shengkun Tang, Yaqing Wang, Caiwen Ding, Yi Liang, Yao Li, and Dongkuan Xu. Deediff: Dynamic uncertainty-aware early exiting for accelerating diffusion model generation. *arXiv preprint arXiv:2309.17074*, 2023. 3

[66] Arash Vahdat, Karsten Kreis, and Jan Kautz. Score-based generative modeling in latent space. *Advances in Neural Information Processing Systems*, 34:11287–11302, 2021. 2

[67] Shuai Yang, Yukang Chen, Luozhou Wang, Shu Liu, and Yingcong Chen. Denoising diffusion step-aware models. *arXiv preprint arXiv:2310.03337*, 2023. 2

[68] Xingyi Yang, Daquan Zhou, Jiashi Feng, and Xinchao Wang. Diffusion probabilistic model made slim. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22552–22562, 2023. 2, 3, 5, 6

[69] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015. 5

[70] Jiahui Yu, Yuanzhong Xu, Jing Yu Koh, Thang Luong, Gunjan Baid, Zirui Wang, Vijay Vasudevan, Alexander Ku, Yinfei Yang, Burcu Karagol Ayan, et al. Scaling autoregressive models for content-rich text-to-image generation. *arXiv preprint arXiv:2206.10789*, 2022. 5

[71] Qinsheng Zhang and Yongxin Chen. Fast sampling of diffusion models with exponential integrator. *arXiv preprint arXiv:2204.13902*, 2022. 2

[72] Hongkai Zheng, Weili Nie, Arash Vahdat, Kamyar Azizzadenesheli, and Anima Anandkumar. Fast sampling of diffusion models via operator learning. In *International Conference on Machine Learning*, pages 42390–42402. PMLR, 2023. 2